Ten Experiments on the Modeling of Polyphonic Timbre

Jean-Julien Aucouturier

THESE DE DOCTORAT DE L'UNIVERSITE PARIS 6
Spécialité: Informatique
présentée par:

Jean-Julien Aucouturier

pour obtenir le grade de
DOCTEUR DE L'UNIVERSITE PARIS 6

Dix Expériences sur la Modélisation du Timbre Polyphonique

Soutenue le 6 Juin 2006.

Devant le jury composé de :

Samy Bengio (IDIAP) - *Rapporteur*
Jean-Pierre Briot (Paris 6, CNRS) - *Co-directeur*
Patrick Gallinari (Paris 6) - *Examinateur*
François Pachet (SONY CSL) - *Co-directeur*
Xavier Rodet (IRCAM) - *Rapporteur*
David Wessel (CNMAT, Berkeley) - *Examinateur*

Résumé

La grande majorité des systèmes d'extraction de metadonnées haut-niveau à partir de signaux musicaux repose sur un modèle implicite de leur "son" ou *timbre polyphonique*. Ce modèle représente le timbre comme la distribution statistique globale d'attributs spectraux instantanés, calculés sur des trames de quelques dizaines de millisecondes. L'hypothèse sous-jacente, rarement explicitée, est que le timbre perçu d'une texture polyphonique correspond à ses attributs instantanés les plus représentés statistiquement. Cette thèse remet en cause la validité de cette hypothèse. Pour ce faire, nous construisons une mesure explicite de la similitude timbrale entre deux textures polyphoniques, déclinée sous un grand nombre de variantes typiques du domaine. Nous montrons que la précision de telles mesures est limitée et que leur taux d'erreur résiduel n'est pas accidentel. Notamment, cette classe de mesures tend à créer de faux-positifs qui sont toujours les même chansons, indépendamment de la requête de départ: des *hubs*. Leur étude établit que l'importance perceptuelle des attributs instantanés ne dépend pas de leur saillance statistique par rapport à leur distribution à long-terme. En d'autres termes, nous "entendons" quotidiennement dans la musique polyphonique des choses qui ne sont pourtant pas présentes de façon significative (statistiquement) dans le signal sonore, mais qui sont plutôt le résultat de raisonnement cognitifs évolués, dépendant par exemple du contexte d'écoute et de la culture de l'auditeur. La musique que nous *entendons* être du piano est surtout de la musique que nous nous *attendons* à être du piano. Ces paradoxes statistico-perceptifs expliquent en grande partie le désaccord entre les modèles étudiés ici et la perception humaine.

Abstract

The majority of systems extracting high-level music descriptions from audio signals rely on a common, implicit model of the global sound or *polyphonic timbre* of a musical signal. This model represents the timbre of a texture as the long-term distribution of its local spectral features. The underlying assumption is rarely made explicit: the perception of the timbre of a texture is assumed to result from the most statistically significant feature windows. This thesis questions the validity of this assumption. To do so, we construct an explicit measure of the timbre similarity between polyphonic music textures, and variants thereof inspired by previous work in Music Information Retrieval. We show that the precision of such measures is bounded, and that the remaining error rate is not incidental. Notably, this class of algorithms tends to create false positives - which we call *hubs* - which are mostly always the same songs regardless of the query. Their study shows that the perceptual saliency of feature observations is not necessarily correlated with their statistical significance with respect to the global distribution. In other words, music listeners routinely "hear" things that are not statistically significant in musical signals, but rather are the result of high-level cognitive reasoning, which depends on cultural expectations, a priori knowledge, and context. Much of the music we hear as being "piano music" is really music that *we expect to be* piano music. Such statistical/perceptual paradoxes are instrumental in the observed discrepancy between human perception of timbre and the models studied here.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

The majority of systems extracting high-level music descriptions from audio signals rely on a common, implicit model of the global sound or *polyphonic timbre* of a musical signal. This model represents timbre as the long-term distribution of the local spectral features, a prototypical implementation of which being Gaussian Mixture Models of Mel-Frequency Cepstrum Coefficients. The underlying assumption is rarely made explicit: the perception of the timbre of a texture is assumed to result from the most statistically significant feature windows.

This thesis questions the validity of this model and this assumption.

To do so, we construct an explicit measure of the timbre similarity between polyphonic music textures, by mobilizing all the tools and design heuristics typically at use in Music Information Retrieval research. We study the properties of the measure in a series of ten experiments.

We show clear evidence that the precision of measures based on this dominant paradigm is bounded by a *glass ceiling* at precision about 70% (**Experiment 1**). The remaining error rate is not incidental, and is indicative of limitations which probably cannot be overcome by variations on the same theme.

One surprising finding of our study is that algorithms that account for the time dynamics of

the features, e.g. with dynamic programming or hidden Markov models, are at best equivalent to simpler static models. This is at odds with experimental data on the perception of individual instrument notes. **Experiment 2** establishes that the polyphonic nature of the data is the main reason that ruins computational attempts at modelling feature dynamics. This suggests that the horizontal coding of frames of data in terms of holistic spectral features, without any account of the synchronicity of sources, is a very inefficient representation of polyphonic musical data, and not cognitively plausible.

An important and novel finding of our study is that the class of algorithms studied in this work tend to create false positives which are mostly always the same songs regardless of the query. In other words, there exist songs, which we call *hubs*, which are irrelevantly close to all other songs. We notably establish that:

- hubs are distributed according to a scale-free distribution.
- hubs are not a consequence of poor feature representation of each individual frame, but rather an effect of the modelling of the agglomeration of the many frames of a sound texture (**Experiment 3**).
- hubs are not a property of a given modelling strategy (i.e. static vs dynamic, parametric vs non-parametric, etc.) but rather tend to occur with any type of model (**Experiment 4**).
- hubs are not an intrinsic property of certain songs, but that different algorithms distribute the hubs differently on the whole database (**Experiment 5**).
- the hubness of a given song is not an emerging global property of the distribution of its frames, but rather can be localised to certain parts of the distribution, notably outlier frames in statistical minority (**Experiment 7**).
- hubs are not a property of the class of algorithms studied here which appears regardless of the data being modelled, but only for data with a given amount of heterogeneity, e.g. for polyphonic music, but not for ecological sound ambiances (**Experiment 8**).

This phenomenon of hubs is reminiscent of other isolated reports in different domains, such as Speaker Recognition or Fingerprint Identification, which intriguingly also typically rely on the same features and pattern-recognition algorithms. This suggests that this could be an important phenomenon which generalizes over the specific problem of timbre similarity, and indicates a general structural property of the class of algorithms examined here. This mostly translates the fact that all data points are not of equal perceptive importance, and that these weights are not necessarily correlated with the statistical significance with respect to the global distribution.

Finally, we give quantitative evidence supporting the fact that "polyphonic timbre" judgements are not low-level immediate perceptions, but rather high-level cognitive reasoning, which depends on cultural expectations, a priori knowledge, and context. **Experiment 9** shows that surprisingly few human-made high-level music descriptions, and notably judgements of instrument classes, are directly correlated to low-level timbre similarity. Polyphonic textures as found in popular music are cultural objects whose perception creates expectations based on music that a particular listener already knows. In **experiment 10**, we show that human judgements could be approximated only by accounting for high-level correlations within a large set of possible categories. Some of these correlations capture psycholinguistical semantic associations ("a powerful song is a strong song"), but also historical and cultural knowledge ("rock uses guitars"), and more subjective aspects linked to perception of timbre ("flute sounds warm").

In other words, music listeners routinely "hear" things that are not statistically significant in musical signals. Such paradoxes associated with polyphonic timbre contribute to the discrepancy between the low-level models of timbre similarity studied in this work and its human perception.

Acknowledgments

This research was conducted from October 2001 to April 2006 in the music team of SONY Computer Science Laboratory, Paris, in academic collaboration with the Laboratoire d'informatique de Paris 6 (LIP6). It was jointly supervised by François Pachet (SONY CSL) and Jean-Pierre Briot

(LIP6), and funded by Association Nationale de la Recherche Technique (ANRT) under a CIFRE grant. It was also partly funded by the two successive European IST projects, Cuidado (IST-1999-20194) and SemanticHifi (IST-2002-2318).

I thank my advisers François Pachet and Jean-Pierre Briot for the trust they put in me, their encouragements and their scientific mentoring. I was immediately impressed and intrigued at François's mention of *Claude Bernard - Introduction à l'Etude de la Médecine Expérimentale*, which presided our first meeting before embarking into all this. After the few years spent here, I still don't think there is another place on earth where that kind of thinking occurs. I hope that this thesis lives up to your expectations and intellectual honesty.

More generally, I thank the many colleagues that I have had at SONY CSL over these years, for the good ambiance and the extremely fertile scientific grounds they provided for my work. In many aspects, this thesis is in direct and humble inheritance of the fascinating depth and width of the scientific culture found in this laboratory. I notably wish to thank Luc Steels, for his constant vision, humanity and availability.

A scary proportion of this work was only made possible by the direct technical participation of other members/affiliates of the Music Team, notably Anthony Beurivé who should be credited for most of the (good) software engineering found here as well as an incredible amount of signal and machine learning implementations, Pierre Roy for combinatorial and search algorithms, Peter Hanappe for system administration, Giordano Cabral, Tristan Jehan and Aymeric Zils for bits and pieces. I thank them for everything.

The final form of this document was influenced by *La Perception de la Musique* by Robert Francès and *Strong Inference* by John Platt, which I encountered thanks respectively to Rémi Gousdoué at EHESS and to an anonymous and absent-minded member of SONY CSL who left his copy at the Xerox machine.

I thank the researchers and professors who accepted to join my thesis committee. I'm honored by their interest in this work. I also thank the members of the my pre-defense committee (Thierry Artieres, Amal El Fallah-Seghrouchni, Geber Ramalho), for their time and opinions. I finally can

nearly reiterate all the acknowledgments that opened my Master's thesis done in Mark Sandler's group while at King's College, University of London, for I am amazed and grateful at the constant kindness and support that my former group has shown ever since. Thank y'all. I hope you'll like it.

But, at the end of the day, all this would not have much sense if it wasn't for Isabelle and Adèle.

Thank you.

Part I

Epistemology

Chapter 2

Dimensions of Timbre

2.1 Everything but pitch and loudness

Timbre is defined by the American Standards Association (ASA, 1960) as "that attribute of sensation in terms of which a listener can judge that two sounds having the same loudness and pitch are dissimilar". In other words, timbre is defined by what it isn't ("everything but"). This *psychoacoustician's waste-basket* denotes notably the quality of a musical note or sound which distinguishes different musical instruments. However, even if we tend to categorize sound sources (e.g. the brass or plucked string musical instrument families), the timbre quality results from an intrinsically non-categorical perceptual process. Sound classes have inner variabilities which makes the space of perceptually meaningful sounds a continuum for which it is difficult to find a perceptual coordinate system. Furthermore, the timbre space is intrinsically multidimensional. Unlike the sensation of pitch which can be fairly directly correlated with the acoustic property of frequency, or the sensation of loudness which correlates to the amplitude, the sensation of timbre is the result of multiple interacting acoustic factors. It is impossible to measure on a single continuum ranging e.g. from the piano to the trumpet.

2.2 Psychophysical studies

A considerable amount of effort has been done in quest for the number of dimensions of a meaningful "timbre space" and for the features of the acoustical signal which best correlate with these dimensions. Most of these studies (Plomp, 1976; Grey, 1977; Wessel, 1991; Kendall and Carterette, 1991; Iverson and Krumhansl, 1993; McAdams et al., 1995) have shared the same methodology: Dissimilarity judgements are collected from a set of human subjects (usually musicians) for a set of individual, natural instrumental sounds spanning different instrument families. In recent studies such as Iverson and Krumhansl (1993); Krimphoff et al. (1994); McAdams et al. (1995), synthetic sounds are also used to create interpolated hybrids between instruments. The similarity ratings are then analysed with Multidimensional scaling (MDS, see e.g. Borg and Groenen (1997)) in order to find a low-dimensional spatial arrangement of the stimuli in a euclidean space such that the distances between data points are optimally respected. Although recent studies such as McAdams et al. (1995) use sophisticated versions of MDS, e.g. with weighted latent classes, in the most simple cases MDS amounts to solving an eigenvalue problem on the distance matrix.

Figure 2.1 shows the timbre space as constructed by MDS in McAdams et al. (1995). Most studies found that the distortion of the collected distance matrix is minimized by a three-dimension arrangement of the data, and give perceptual or acoustic interpretations for each dimension. The most important dimension has usually been consensually (Grey, 1977; Wessel, 1991; Kendall and Carterette, 1991; Iverson and Krumhansl, 1993; Krimphoff et al., 1994; McAdams et al., 1995) correlated to the centroid of the spectral envelope, which measures the spectral energy distribution in the steady state portion of a tone, and corresponds to perceived "brightness". The second dimension is generally associated with the log of the attack-time, i.e. the time between the onset and the instant of maximal amplitude. However, this feature was criticized by Iverson and Krumhansl (1993), who observed that the fact of including or deleting the attack portion of the stimuli had little influence on the perceptual structure of timbres found by MDS.

The final dimension of the 3D timbre space has been much debated, and been associated to

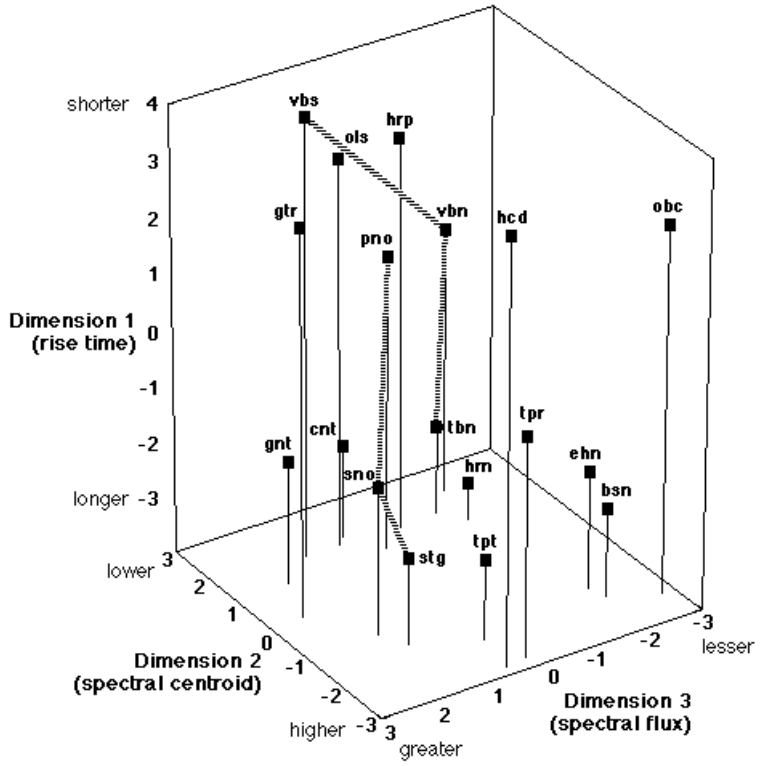

Figure 2.1: 3D timbre space derived from dissimilarity ratings on 18 timbres by 88 subjects. The acoustic correlates of the perceptual dimensions are indicated in parentheses. Hashed lines connect hybrid timbres (*vibrone* and *striano*) to their progenitors (vibraphone/trombone and bowed string / piano, resp.). Reproduced from McAdams et al. (1995)

diverse temporal and spectral features depending on the author: spectral variation over time (Grey, 1977; Iverson and Krumhansl, 1993), spectral irregularity ("log of the standard deviation of component amplitudes from a global spectral envelope derived from a running mean of the amplitudes of three adjacent harmonics", Krimphoff et al. (1994)), Spectral Flux (average of the correlations between amplitude spectra in adjacent time window, McAdams et al. (1995)). The unconvincing correlation scores achieved by the various proposed features probably simply indicate that this last dimension corresponds to a residual error of the assumption made by MDS of a euclidean embedding of the data. This is further confirmed by McAdams et al. (1995), who identify several specificities of individual timbres that explain some large deviations from the ideal euclidean model. Such

specificities can be either continuous attributes, such as the “raspiness of the attack” (a specificity of trombone sounds), or discrete properties, such as “a suddenly pinched offset with a clunk” (for harpsichord sounds). These specificities, while not accounted for in the euclidean model because of their locality to individual timbres, are sufficiently salient to influence the measured dissimilarity of some timbres.

Several studies have further tried to recreate meaningful timbre space by testing various distance measures computed on the signals. Plomp (1976) successfully recreates the same spatial structure as with human similarity ratings by using distance measures computed on the vector of energy levels in an auditory filterbank, which gives further support in favour of the perceptual predominance of factors related to the spectral envelope. Recent studies (DePoli and Prandoni, 1997; Terasawa et al., 2005) have proved that Mel-Frequency Cepstrum Coefficients (MFCC, see Chapter 3.1), a particular encoding of the spectral envelope widely used in the speech recognition community, provide a good distance that reproduces timbre clusters that are similar to previous MDS studies (DePoli and Prandoni, 1997) and account for 66% of the perceptual variance in human timbre similarity judgements for steady state, individual sounds (Terasawa et al., 2005).

2.3 Automatic recognition of monophonic timbres

The psychophysical studies on the perceptive dimensions of musical timbre have provided motivation to the automatic computer recognition of musical instruments from audio samples corresponding to individual notes (Herrera-Boyer et al., 2003). While early systems tend to use either spectral information (Brown, 1997) or temporal information (Martin, 1998), most studies since Eronen and Klapuri (2000) rely on a combination of both aspects. Spectral correlates of timbre are often measured with Mel-Frequency Coefficients, which are generally found to be the most important features. Brown (1997) reports near-perfect recognition performance on a small set of saxophone and oboe sounds, using MFCCs only (with simple gaussian classifiers). On a much larger problem (1500 samples covering the entire pitch ranges of 30 orchestral instruments), Eronen (2001) also identifies

MFCCs as the best individual classification feature, accounting for 32% precision for individual instruments (65% for instrument families) when used alone (out of a combined best precision of 35% (77%)). The contribution of the MFCC feature can be compared with 10% (35%) precision achieved by the second-best feature found by Eronen and Klapuri (2000), the standard deviation of the spectral centroid.

Temporal features however are generally found necessary to combine to spectral cues in order to reach acceptable recognition performance. While most psychoacoustical studies typically propose to compute them based on the simple energy envelope of the signal, most automatic recognition systems propose to base them on relatively complex signal analysis. Martin (1998) calculates a set of temporal attributes such as vibrato frequency, tremolo, centroid modulation frequency, slope of onset harmonic skew, etc. on the outputs of a log-lag correlogram, which measures the auto-correlation with logarithmic lag in each frequency band of a gammatone filterbank, modelling the frequency resolution of the cochlea (Smith and Abel, 1999). Eronen (2001) base its computations on the matrix of harmonic amplitude envelopes, obtained from a partial tracking algorithm in each band of a Bark filterbank. The amplitude envelopes are then analysed to extract a number of features such as band-wise rise time, mean, variance, strength and frequency of amplitude modulation, etc.

Most features extracted from individual, mono-instrument notes can be combined into a unique feature vector for each sample. The feature vectors corresponding to each note can then be compared to one another using simple metrics such as Euclidean distance:

$$d(s_i, s_j) = \sum_{k=0}^{N-1} |f_k(s_i) - f_k(s_j)| \tag{2.1}$$

where s_i and s_j T and $f_k(s)$ is the value of the k^{th} feature for sample s. The classification is then typically done using the K-Nearest Neighbors algorithm (Fujinaga, 1998; Martin, 1998; Eronen and Klapuri, 2000). The algorithm first stores the feature vectors of all the training examples and then, for classifying a new instance, finds a set of k nearest training examples in the feature space, and assigns the new example to the class that has more examples in the set. However, more complex

density estimation methods such as Gaussian Mixture Models (Brown, 1997) and classifiers such as Support Vector Machines (Essid et al., 2004) are often used in an attempt to better models the decision boundaries between classes. Finally, a number of contributions have emphasized the need of modelling the dynamic trajectory of local features, relying on complex dynamical models such as hidden Markov models or recurrent neural networks. Dubnov and Fine (1999) propose to consider the sound dynamics as a stochastic process over the feature domain, and to match instrument samples by estimating cross entropy between the corresponding stochastic models. Eronen (2003) proposes to use 3-state left-right hidden Markov models, to model the succession of different feature distributions for onset, steady state and decay. This is motivated e.g. by the observation that some instruments are characterized by onset asynchrony, which means that the energy of certain harmonics rises more quickly than the energy at some other frequencies.

2.4 Towards polyphonic textures

Most of the studies on musical instrument recognition have focused on sound samples corresponding to clean recordings of a unique note, played by a single instrument, which is an unrealistic context for possible music applications. On the one hand, the music industry are in demand of models of timbre perception that are suitable for real-world, complex polyphonic textures of several seconds' length. On the other hand, the approach of monophonic instrument recognition seems little suited to the modelisation of such complex textures.

2.4.1 The demand of Electronic Music Distribution

The exploding field of Electronic Music Distribution (EMD, Pachet (2003)) is in need of powerful content-based management systems to help the end-users navigate huge music title catalogues, much as they need search engines to find web pages in the Internet. Not only do users want to find quickly music titles they already know, but they also — and perhaps more importantly — need systems that help them find titles they do not know yet but will probably like.

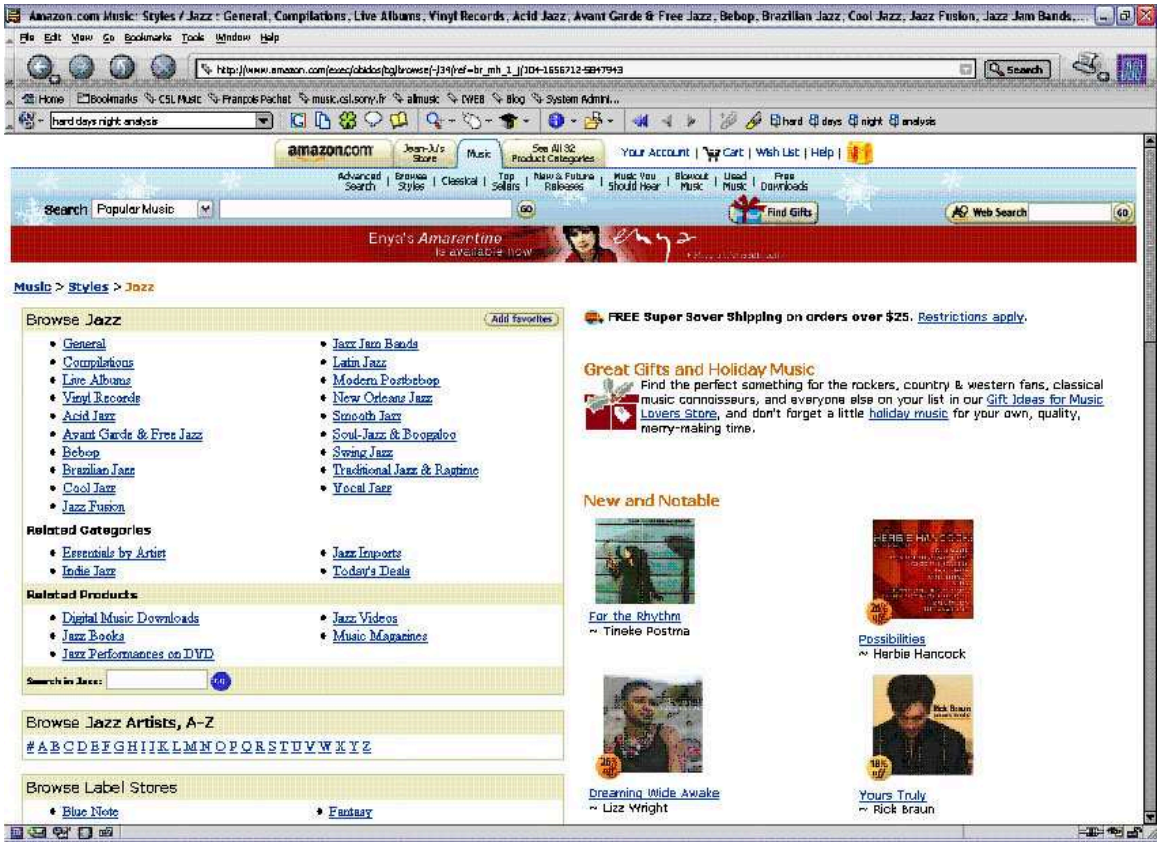

Figure 2.2: A typical EMD system, using predefined genre / style taxonomies as unique content-based access mode: web music-seller Amazon.com

Many content-based techniques have been proposed recently to help users navigate in large music catalogues. The most widely used is collaborative filtering. This technique is based on the analysis of large numbers of user profiles. When patterns are discovered in user profiles, corresponding music recommendations are issued to the users. Systems such as *Amazon.com* exploit these technologies or variants thereof (Pachet et al., 2001; French and Hauver, 2001; Pestoni et al., 2001) with various degrees of success.

The main drawback of these approaches is that they are essentially content-blind; the music itself is ignored, and only users tastes are considered. The resulting recommendations are therefore at best superficially relevant (see Figure 2.2). Other content-based management techniques attempt

at extracting information directly from the music signal. In the context of Mpeg7 in particular, many works have addressed the issues of extracting automatically features from audio signals, such as tempo (Scheirer, 1998), rhythm or melodies (Gómez et al., 2003). The resulting descriptors can be used for querying music catalogues by content information rather than by song or artist names, and as such provide a first layer to content-based music access. Query by humming is probably the most spectacular of these approaches (Ghias et al., 1995). However, these are limited essentially by the difficulty for non-specialists to identify the right descriptors. Query by humming for instance, is largely dependent of the ability of the user to sing correctly a song. Furthermore, these techniques by construction only help users to find what they actually look for, and therefore address only a small fraction - and the easiest one - of the EMD problem.

In this context, the global sound or timbre of a polyphonic texture seems an ideal candidate to build successful EMD systems. Although it is difficult to define precisely music taste, it is quite obvious that music taste is often correlated with timbre. Some sounds are pleasing to listeners, other are not. Some timbres are specific to music periods (e.g. the sound of Chick Corea playing on an electric piano), others to musical configurations (e.g. the sound of a symphonic orchestra). In any case, listeners are sensitive to timbre, at least in a global manner, as confirmed by an increasing amount of user studies of recommendation systems and music libraries (Baumann et al., 2004; Lee and Downie, 2004).

Moreover, timbre similarity is a very natural way to build relations between music titles. The very notion of two music titles that sound the same seems to make more sense than, for instance, query by humming. Indeed, the notion of melodic similarity is problematic, as a change in a single note in a melody can dramatically impact the way it is perceived (e.g. change from major to minor). Conversely, small variations in timbre will not affect the timbral quality of a music title, considered in its globality.

Figures 2.3 to 2.6 illustrate possible examples of similar “polyphonic timbres” as we understand it here.

Figure 2.3: Timbre similarity between a bossa-nova piece by *Joao Gilberto* (left) and a folk tune by English songwriter *Bert Jansch* (right): both consist of a simple acoustic guitar and a gentle male voice.

2.4.2 The lack of perceptive models

On the other hand, both the conclusions of psychophysical experiments on monophonic timbre perception and the approach of monophonic instrument computer recognition seem little suited to the modelisation of such complex polyphonic timbres, for a number of reasons we list here.

- Polyphony: Kendall and Carterette (1991) is, to the best of our knowledge, the only study documenting the timbre perception of mixtures of multiple instrument. The authors have collected human dissimilarity judgements for dyads of instruments playing either single tones (at unison or distant of a major third interval) or simple melodies (again, at unison and in harmony), and compared the data with the collected distances for the individual instruments composing the dyads. They found that, to a limited extent, a quasi-linear vector model could explain the perception of timbre combinations on the basis of the vector sum of the positions of the constituent timbres. This suggests that attributes of timbre inferred by such studies such as spectral centroid or amplitude variations are perceived for linear combinations of sounds as the summation of the individual sound attributes. However, many features proposed for individual sounds are not linear functions of the signal, especially temporal descriptors such as energy variations or rise time, but also e.g. RMS-energy in frequency bands. Therefore

Figure 2.4: Timbre similarity between a song by *the Rolling Stones* (left) and by *the Beatles* (right): both songs have the sound signature of 1970's English pop-rock, with lead, rhythm guitars, bass and drums, and often doubled vocals.

their computed value for mixtures of signals is not a linear combination of their individual values. This suggests that, even if still conceptually valid for polyphonic timbres, the features usually computed for monophonic signals cannot be directly applied to polyphonic signals. They would for instance would require some kind of source separation of the individual components, which is still a difficult research problem of its own (Plumbley et al., 2002). Moreover, the findings of Kendall seem at odd with psychoacoustical data on spectral masking, according to which the perceived interaction of sound sources is not the linear sum of the individual components. This effect can be modeled by a spreading function such that frequency components in a given frequency band contribute to neighboring bands proportionally to their distance. Figure 2.7 shows a schematic representation of spectral masking, where the sinusoidal tone *C* is masked by the excitation pattern of the neighboring, higher-amplitude tone *B*. The formulation of a spreading function optimized for speech signals can be found in Schroeder et al. (1979). Pampalk et al. (2003) and Lidy and Rauber (2005) have notably studied its application to music signals.

- Asynchronicity: Different sound sources in a musical mixture are typically not synchronized (except e.g. when playing at unison). A sound segment corresponding to a given note of a

Figure 2.5: Timbre similarity between a jazz piece by *Charlie Haden*'s 1976 band *Old and New Dreams* (left) and *Charles Mingus*' 1956 sessions on Debut (right), each with a bass-player leader and dense collective textures from the brass section (saxophone alto, tenor and trumpet).

given instrument is likely to be superimposed with other notes with various time offsets. For instance, the attack of a piano note extracted from the recording of a jazz trio may superimpose with the decay of a double-bass note, and its steady-state may be similarly corrupted by several drum onsets. This makes time descriptors such as log-attack time quasi-impossible to compute meaningfully for polyphonic sound mixtures. Moreover, the large variability of offsets between events of individual sound sources makes it very difficult to learn feature dynamics with e.g. HMMs or Markov chains as proposed by Dubnov and Fine (1999): an extremely large number of training instances would be needed to describe all possible degrees of time superpositions of individual notes (drum onset at 10% of piano steady state, drum onset during piano attack, piano attack during drum decay, etc.).

- Auditory Stream Segregation: Wessel (1991) and Singh and Bregman (1993) have demonstrated the mutual perceptual influence of timbre and phrase grouping. The way a succession of notes is perceived and grouped into phrases depends on the timbre of the notes. Notes with similar timbres are typically grouped together, and the perception of simple periodic patterns may be altered by introducing abrupt timbral changes (Figure 2.8). Reciprocally, two timbres may be judged differently depending on the metrical function and context of the correspond-

Figure 2.6: Timbre similarity between the *Spice Girls* (left) and the *All Saints* (right), two British pop acts of the mid-'90s: 4 female R&B vocalists and catchy dance-pop back-up reminiscent of *Madonna*.

ing notes. This suggests that polyphonic textures of several notes (all the more so several seconds) may be perceived in a different way than the majority perception of each individual notes.

- Noise: Few psychoacoustical studies so far have documented the timbre perception of noisy sounds. The MDS experiments of Grey (1977) identified high frequency noise when preceding the attack as an important attribute, but it was later discarded in Iverson and Krumhansl (1993). One possible explanation is the lack of representations and analysis techniques for noise sounds, which cannot be described in the framework of steady-state harmonic sounds, although recent work such as Hanna and Desainte-Catherine (2003) makes this perspective more of a reality.

- From timbre to sound: The timbre of real-world polyphonic textures cannot be reduced to the superposition of clean individual timbres of the component instruments. Contemporary popular music (possibly since the 1960's) is the result of applying many production techniques on the original sound captures, such as sound effects (reverberation, chorus, :.), informed choices of sound recording equipments (e.g. lamp amplifiers) and sound engineering practices. Certain types of music production have often gained a notoriety of their own, e.g. the *"Phil*

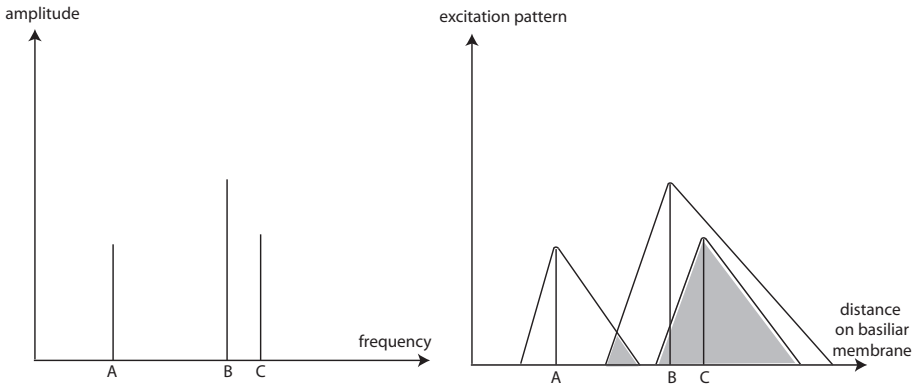

Figure 2.7: Illustration of the spectral masking effect. The stimulus (left) is composed of three sinusoidal tones, presented simultaneously. The auditory representation of the stimulus (right) shows the excitation pattern provoked by each tone on the basiliar membrane. The spreading of the excitation provoked by sinewave *B* masks the perception of *C*, but not the perception of *A*. Note that lower frequencies have a stronger masking influence on higher frequencies than vice versa.

Spector sound" (Figure 2.9), characterized by the so-called "Wall of Sound", a production technique that involved very many musicians playing at unison, yielding a dense and layered effect that was highly influential on 1960's and 1970's pop music, including *The Beatles* and *The Ramones* (Ribowski, 1989). Similarly, ECM Jazz label's motto, the "most beautiful sound next to silence", illustrates a concern for a certain quality of sound recording, which had an incredible impact on jazz (Delalande, 2001). While extremely salient (positively or negatively) for the music lover, this transition from (multiple) timbres to a global "sound" of music has seen few analytical studies, with the notable exception of Berger and Fales (2003) who investigate the acoustical correlates of the "heaviness" of electric guitar sounds in *Heavy Metal* music, and conclude that 'heavier" timbres correlate with a gradual fading of high-frequency energy and flatter dynamic envelop.

- Semantic Textures: Polyphonic textures of several seconds' length are a difficult material for timbre perception studies and computational models because it is generally difficult to abstract their global timbre/sound percept from higher-level concepts such as music genre, style, tastes, mood to name but a few. Moreover, while intuition may hint at the contrary, such con-

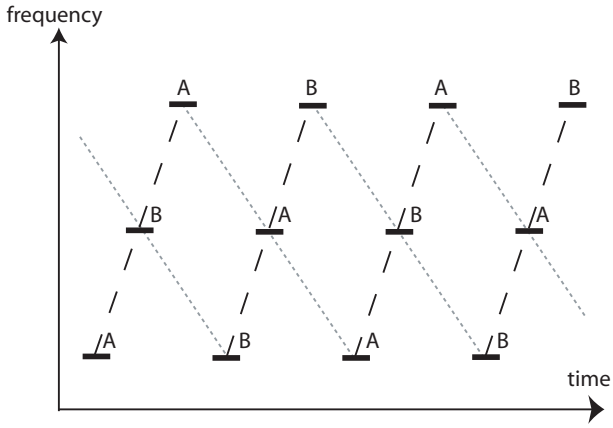

Figure 2.8: Classical Experiment illustrating the influence of timbre on stream segregation. A series of patterns of 3 ascending notes is presented to the subject. When the timbre distance between adjacent notes is small, the repeating ascending pitch line (dark-gray dashed) dominates the perception. If the distance between timbres of *A* notes and *B* notes increases, the notes are grouped by timbre similarity and 2 interleaved descending pitch lines are heard (light-gray dotted) (Wessel, 1991).

cepts lack any kind of semiology or theory to explain their relation to musical sound. Musical genre, for instance, is a ill-defined concept, which is reflected by the inconsistencies of existing genre taxonomies. Pachet and Cazaly (2000) compare 3 Internet genre taxonomies: allmusic.com (531 genres), amazon.com (719 genres) and mp3.com (430 genres). Results show that there is no consensus in the names used in these classifications: only 70 words are common to the three taxonomies. More importantly, there are no shared definitions: among these common words, even largely used terms like Rock or Pop do not denote the same set of songs. Typical genre definitions reflect dimensions of music such as timbre, harmony or rhythm (the main difference between *reggae* and *ska* for instance is the tempo), but also and foremost cultural interpretations, which depend on epochs, locations and user communities (Aucouturier and Pachet, 2003).

Figure 2.9: Harvey Phillip ”Phil” Spector (b. 1940), a highly influential American record producer and inventor of the “Wall of Sound” production technique, which became the signature sound of 1970’s rock and roll music.

2.5 Implicit modelling

The lack of psychophysical models for the timbre perception of polyphonic textures has lead researchers to take a pragmatic approach to build the much-needed automatic systems able to extract high-level descriptions (HLD) of music signals (such as genre). The approach, based on pattern recognition (Bishop, 1995), is a direct extension of the most simple monophonic instrument recognition systems. The signal is cut into short overlapping frames (typically 50ms with a 50% overlap), and for each frame, a feature vector is computed. Features usually consists of a generic, all-purpose spectral representation such as Mel Frequency cepstrum Coefficients (MFCC), since more complex temporal correlates of timbre identified by psychophysical studies are ill-defined in the case of polyphonic textures, as seen above. The timbre of individual sound samples is not explicitely modelled: all feature vectors are fed to a classifier which models the global distributions of the features of

Table 2.1: Number of contributions using the timbre paradigm in the past ISMIR symposiums

year	timbre papers	total papers	percentage
2000	6	26	23%
2001	9	36	25%
2002	14	58	24%
2003	12	50	24%
2004	23	104	22%
2005	24	114	21%
total	88	388	23%

signals corresponding to each class (e.g. rock or jazz in the case of a genre classification system). Global distributions for each class can the be used to compute decision boundaries between classes. A new, unobserved signal is classified by computing its feature vectors, finding the most probable class for each of them, and taking the overall most represented class for the whole signal.

This approach for modelling HLD concepts correlated to the global sound of a musical extract is a widely adopted paradigm in the research community concerned with automatic music description. Table 2.1 shows an enumeration of paper and poster contributions in the ISMIR conferences since its creation in 2000. Each year, about a fourth of all papers, and on the whole 88 papers out of a total 388, use the approach. Each contribution typically instantiates the same basic architecture described above, only with different algorithm variants and parameters.

All contributions use the same underlying rationale of modelling global timbre/sound in order to extract high-level descriptions. However the spectrum of the targeted descriptions is rather large: genre (Tzanetakis et al., 2001), style (Whitman and Smaragdis, 2002), mood (Liu et al., 2003), speech/music (Scheirer and Slaney, 1997), solo instrument (Vincent and Rodet, 2003), singer (Kim and Whitman, 2002), but also singing language (Tsai and Wang, 2003), type of beat-boxing sounds (Kapur et al., 2004), potential for commercial success (Dhanaraj and Logan, 2005), etc. Note that while the “global timbre” paradigm is by far the most represented for HLD extraction systems, several other approaches exist that rely e.g. on harmonic features such as pitch histograms (Ermolinskiy et al., 2001), rhythmic features (Gouyon and Dixon, 2004) or cultural information mined from web pages (Whitman and Smaragdis, 2002).

These contributions rely on an implicit model of global timbre, which is based on the long-term statistics of frame-based spectral features. The underlying assumption is that the perception of the timbre of a texture is assumed to result from the most statistically significant feature windows.

However, the validity of such a model and assumption is difficult to discuss from the existing corpus of work, because is not examined explicitly:

- Negative results are inconclusive: The evaluation of each contribution does not measure the precision of the decisions of the models in terms of timbre similarity, but in terms of complex high-level concepts which may be only remotely correlated to timbre. For instance, the difficulty of a specific genre classification, e.g. between “Classical” and “Jazz”, may reveal limitations of the underlying timbre model, but also possible inconsistencies in the measured concepts: *Carla Bley* or *Gil Evans*’ complex orchestral jazz arrangements are timbrally more similar e.g. to *Alban Berg* than to *Charlie Parker*. Hence, it is difficult to conclude either on the quality of the timbre model or on the relation between timbre perception and the modelled high-level concept.

- Positive results are inconclusive: HLD extraction systems all rely on classification techniques which exploit the distribution of the data to discriminate sets of sounds. The performance of such systems tell very little on the underlying model of timbre similarity, as it is possible to model appropriate decision boundaries on a feature space which does not provide a precise model of similarity, thanks to effective use of training data. Moreover, modern classification techniques such as ensemble learning (McKay et al., 2005) have the ability to select different decision criteria according to sub-parts of the problem, e.g. different duplets of classes being discriminated. Such classifiers are typically able of good classification results on feature spaces which would give a poor metric representation of the underlying similarity.

- Individual evaluations are inconclusive: The improvement of HLD extraction systems is actively pursued in the MIR community. Each contribution typically relies on a slightly modified timbre model, with its own algorithm variants and parameter settings. The choice of

such parameters result from little if any systematic evaluation. More generally, attempts at evaluating different settings in the literature tend to compare individual contributions to one another, i.e. particular, discrete choices of parameters, instead of directly testing the influence of the actual parameters. Moreover, each contribution typically measures the performance of its particular implementation on its own custom-built testing database, which makes the comparison of different approaches difficult and unreliable. It is very common that good results found on a given test database cannot be reproduced by subsequent studies on other databases, either quantitatively or even qualitatively. For instance, Soltau (1998) reports that Explicit-Time Modelling with Neural Network (a technique to model the dynamics of the features) significantly outperforms hidden Markov models on a genre classification task, which does not occur in Scaringella and Zoia (2005).

2.6 Thesis Overview: Ten Experiments

In this context, we propose to *explicitly* model polyphonic timbre, by building a algorithmic measure of the *timbre similarity of polyphonic textures*, much like the similarity assessed by psychophysical experiments on short samples. Our measure is based on the pattern recognition approach shared by most HLD extraction systems, namely modelling polyphonic timbre as the long-term distribution of local spectral features.

By focusing on the low-level perceptive mechanism of timbre similarity, we aim at studying the validity of the approach shared by the contributions described above, without depending on the unknown correlations that exist at the level of high-level music descriptions. We study the properties of such models of timbre similarity in a set of 10 experiments, whose conclusions have implications on polyphonic timbre, but also on high-level music descriptions built on the timbre rationale. Part of our results (the existence of so-called hub songs) even seem generalize to pattern-recognition measures in general.

Each contribution in the literature typically instantiates the same basic architecture described

above, only with different algorithm variants and parameters. Chapter 3, **Dimensions of Timbre *Models***, gives a detailed description of the space of such timbre models, in terms of a prototypical center-of-mass algorithm ("gaussians of MFCCs"), to which we apply a number of transformations (or so-called *design patterns*) which we found to re-occur very frequently in the literature (for instance, transformations meant to better model the dynamics of the features). This review sketches a practical *epistemology* of the field of music pattern recognition, by making more explicit the typical signal-processing and machine-learning heuristics and know-how at use among its practitioners*. It also provides a structured way to explore the space of all possible timbre models, on which we base our experiments in the remainder of the study.

Experiment 1 tests the validity of the assumptions underlying the pattern recognition approach to polyphonic timbre, by hill-climbing the algorithmic space described in Chapter 3. In particular, we question the common assumption found in the literature that error rates reported on implicit models of timbre are incidental, and that near-perfect results would just extrapolate by fine-tuning the algorithms parameters. We propose an evaluation framework which targets explicitly the notion of timbre similarity, instead of derived high-level descriptions, and uses this framework to evaluate the precision of very many parameters and algorithmic variants, some of which have already been envisioned in the literature, some others being inspired from typical patterns of research methodologies observed in the literature. This leads to an absolute improvement over existing algorithms of about 15% precision. But most importantly, we describe many variants that surprisingly do not lead to any substancial improvement of the measure's precision. Moreover, our simulations suggest the existence of a *glass ceiling* at precision about 65% which probably cannot be overcome by pursuing such variations on the same theme.

Experiment 2 further examines one of the most surprising results of **Experiment 1**, namely that algorithms that account for the time dynamics of the features are at best equivalent to simpler static models. This contradicts experimental data on the perception of individual instrument notes, which established the importance of dynamics, notably the attack time and fluctuations of the spectral

*this reflexion on research heuristics used in MIR is largely influenced by the idea behind the automatic discovery system EDS (Zils and Pachet, 2004)

envelope. We propose that the difficulty of modelling dynamics of full songs results either from the complex structure of the temporal succession of notes, or from the vertical polyphonic nature of individual notes. We discriminate between both hypothesis by comparing the performance of dynamical algorithms on several specially designed datasets, namely monophonic individual notes, polyphonic individual notes, and polyphonic multiple-note textures. We conclude that the main cause of the difficulty of modelling dynamics is the polyphonic nature of the data.

The other important and novel finding of **Experiment 1** is that the class of algorithms studied in this work tend to create false positives which are mostly always the same songs regardless of the query. In other words, there exist songs, which we call *hubs*, which are irrelevantly close to all other songs. This phenomenon is reminiscent of other isolated reports in different domains, such as Speaker Recognition or Fingerprint Identification, which intriguingly also typically rely on the same features and pattern-recognition algorithms. This suggests that this could be an important phenomenon which generalizes over the specific problem of timbre similarity, and indicates a general structural property of the class of algorithms examined here. **Experiments 3** to **8** aim at better understanding the nature and causes of such hub songs.

Experiment 3 establishes that hubs are not a consequence of poor feature representation of each individual frame, but rather an effect of the modelling of the agglomeration of the many frames of a sound texture.

Experiment 4 shows that hubs are not a property of a given modelling strategy (i.e. static vs dynamic, parametric vs non-parametric, etc.) but rather tend to occur with any type of model.

Experiment 5 shows that hubs are not an intrinsic property of certain songs, but that different algorithms distribute the hubs differently on the whole database.

Experiment 6 disproves the hypothesis that hubs result from the fact that a given statistical model potentially explains very many different time series of features (for instance, static models such as Gaussian Mixture Models consider all permutations of the original audio data as identical). Notably, we establish that songs whose models have the greatest variance are not significantly likely to be hubs.

Experiment 7 suggests that the hubness of a given song is not an emerging global property of the distribution of its frames, but rather can be localised to certain parts of the distribution. Notably, not all frames have a uniform influence on the hubness of a song. Outlier frames in statistical minority seem to have a critical influence on the appearance of a hub.

Experiment 8 finally establishes that hubs are not a property of the class of algorithms studied here which appears regardless of the data being modelled. Notably, the same algorithms used on datasets of ecological urban sound ambiances do not engender hubs. A possible critical factor for the appearance of hubs is the heterogeneity of the modelled signals, which is 3-4 times higher for polyphonic music than for ecological sounds.

The two experiments which conclude this study make a round trip back to high-level music descriptors. Having explicitly evaluated the validity of polyphonic timbre models and studied a number of their limitations, we now examine the validity of their use to extract conceptual information such as musical genre. We base our study on a yet-unreleased very large and diverse set of manually collected metadata, made available to Sony CSL by collaborations with the Sony Corporation. **Experiment 9** shows that surprisingly few high-level music descriptors are directly correlated to timbre. Moreover, different taxons of a given category, such as “Mood Violent” or “Mood Ironical”, have very diverse levels of correlation with timbre (high and low resp.), which is at odds with typically proposals of classifiers that apply the same decision space for every taxons within a category.

However, **Experiment 10** shows that there are extreme amounts of correlation between high-level descriptors, independently of their relation to timbre. Some of these correlations capture psycholinguistical semantic associations (“a powerful song is a strong song”), but also historical and cultural knowledge (“rock uses guitars”), and more subjective aspects linked to perception of timbre (“flute sounds warm”). This suggests that very many high-level cultural descriptions of music *can* indeed be grounded to timbre similarity, by exploiting such higher-level correlations with timbre-based attributes. We finally propose a hybrid classification system that implements this idea in a systematic way.

Chapter 3

Dimensions of *Timbre Models*

Very many contributions in automatic music description systems rely on the same implicit model of polyphonic timbre, namely the long-term distribution of local spectral features. Most contributions instantiate the same algorithmic architecture, only with a wealth of individual variants and parameter settings. This chapter describes this space of algorithm parameters and variants. We base our description on a prototypical center-of-mass algorithm ("gaussians of MFCCs"), to which we propose to apply a number of transformations (see Figure3.1).

We propose a classification of such transformations in terms of *design patterns* (in the spirit of Gamma et al. (1995)), which we found to re-occur very frequently in the literature (for instance, transformations meant to better model the dynamics of the features). These design patterns make explicit the heuristics and know-how at use among signal-processing and machine learning specialists, and constitute a first step toward sketching a practical *epistemology* of the growing field of Music Information Retrieval (MIR).

The description found below also provides a basis for the experiments in the remainder of our study. Notably, in the next chapter, **Experiment 1** will examine the validity of the assumptions behind all such possible timbre-models, by hill-climbing the algorithmic space described here.

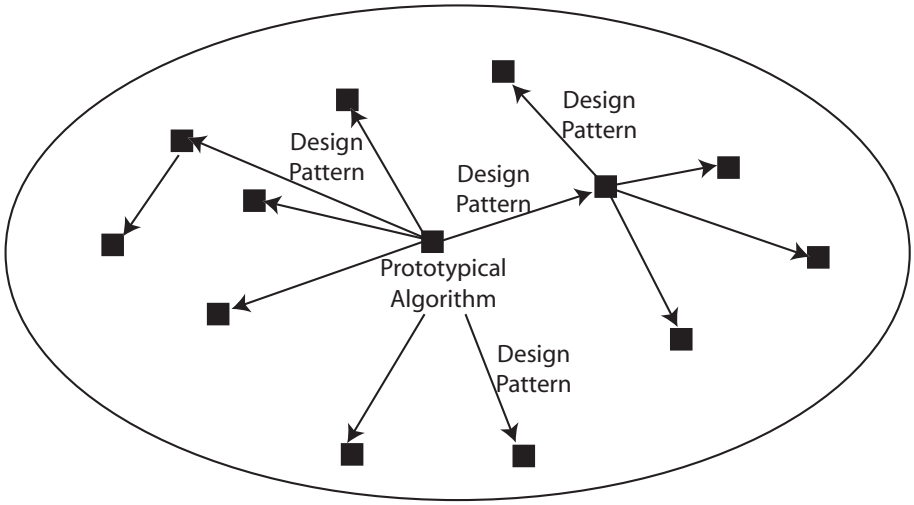

Figure 3.1: We propose to base our exploration of the space of all possible timbre models on a prototypical algorithm, to which we apply a number of transformations, or *design patterns*.

3.1 The Prototypical algorithm

We describe here the timbre similarity algorithm on which we will base our experimentations. As can be seen in Figure 3.2, it follows the same paradigm as the very many contributions on HLD extraction systems described in Chapter 2.5, namely modelling polyphonic timbre as the long-term distribution of local spectral features. However, the rationale behind our experiments is to explicitly model timbre, instead of using it implicitly to extract higher-level, correlated concepts. Therefore, we do not accumulate all features of the different sets of songs to discriminate in a common classifier, but define a metric to compare the distributions of individual songs to one another.

3.1.1 Feature Extraction

The signal is first cut into frames. For each frame, we estimate the spectral envelope by computing a set of Mel Frequency Cepstrum Coefficients. The cepstrum is the inverse Fourier transform of the log-spectrum $\log \mathcal{S}$.

$$c_n = \frac{1}{2\pi} \int_{\omega=-\pi}^{\omega=\pi} \log \mathcal{S}(\omega) \exp\, j\omega n \, \mathrm{d}\omega \tag{3.1}$$

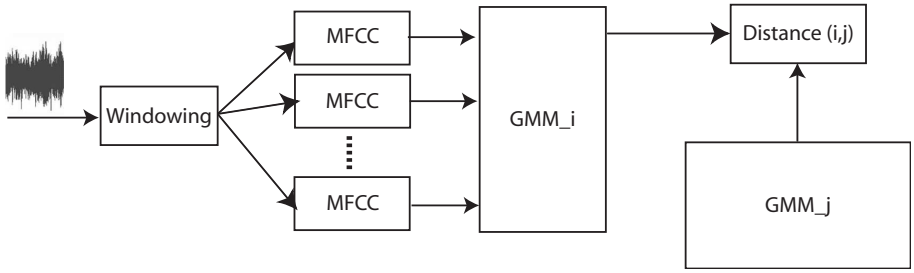

Figure 3.2: The initial algorithm has a classical pattern recognition architecture.

We call mel-cepstrum the cepstrum computed after a non-linear frequency warping onto a perceptual frequency scale, the Mel-frequency scale (Rabiner and Juang (1993)), which reproduces the non-linearity of the frequency resolution of the human auditory system (low Hertz frequencies are more easily discriminated than high Hertz frequencies). Hertz frequency f can be converted to mel frequency m using the experimental formula :

$$m = 1127.01048 \ln (1 + \frac{f}{700}). \tag{3.2}$$

The c_n in Equation 3.1 are called Mel frequency cepstrum coefficients (MFCC), of which we keep a given number N. Cepstrum coefficients provide a low-dimensional, smoothed version of the log spectrum (all the more so precise as N increases), and thus are a good and compact representation of the spectral shape. As already mentioned, they are widely used as feature for speech recognition, and have also proved useful in musical instrument recognition (Eronen and Klapuri, 2000).

3.1.2 Feature Distribution Modelling

We then model the distribution of the MFCCs over all frames using a Gaussian Mixture Model (GMM). A GMM estimates a probability density function (PDF) as the weighted sum of $\mathcal{M}$ simpler

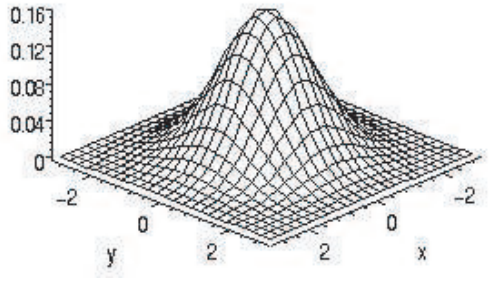

Figure 3.3: A Gaussian distribution in dimension $d = 2$ centered on the origin and with unitary variance in each dimension.

Gaussian densities, called components of the mixture. (Bishop (1995)):

$$p(\mathcal{F}_t) = \sum_{m=1}^{m=\mathcal{M}} w_m \mathcal{N}(\mathcal{F}_t, \mu_m, \Sigma_m) \tag{3.3}$$

where w_m is a mixture coefficient (also called component weight or prior probability), $\mathcal{F}_t$ is the feature vector observed at time t, and $\mathcal{N}$ is a Gaussian PDF with mean μ_m and covariance matrix Σ_m :

$$\mathcal{N}(\mathcal{F}_t, \mu_m, \Sigma_m) = \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma_m|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x - \mu_m)^T \Sigma_m^{-1}(x - \mu_m)\right) \tag{3.4}$$

where d is the dimension of the feature vector $\mathcal{F}_t$. Figure 3.3 shows a 3D representation of a Gaussian distribution in dimension $d = 2$, with $\mu = [0, 0]$ and $\Sigma = \mathcal{I}$, and Figure 3.4 shows a mixture model of $M = 2$ gaussian components ($w_1 = w_2 = 1$, with $\mu_1 = [0, 0]$ and $\mu_2 = [3, 0]$).

The parameters of the GMM are estimated from the feature vectors with the classic E-M algorithm (Bishop (1995)).

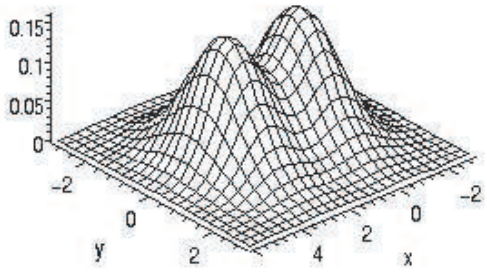

Figure 3.4: A Gaussian mixture model distribution in dimension $d = 2$, with 2 gaussian components centered in $[0, 0]$ and $[3, 0]$.

3.1.3 Distance Measure

Log-likelihood

We can now use these GMMs to match the timbre of different songs, which gives a similarity measure based on the audio content of the music. The most straightforward way to compare two songs A and B on the basis of their respective distributions of feature vectors is to compute the probability of observing the set of all S_A feature vectors of A using the probability model of the distribution of the features of B, i.e.

$$p_B(A) = p_B(\mathcal{F}_1^A, \mathcal{F}_2^A, \ldots, \mathcal{F}_{S_A}^A) \quad (3.5)$$

$$= \prod_{i=1}^{S_A} p_B(\mathcal{F}_i^A) \quad (3.6)$$

where $p_B(\mathcal{F}_i^A)$ is the probability of the i^{th} feature vector of song A computed using Equation 3.3, using the parameters μ, Σ and w learned from the feature vectors of B. To avoid double-precision numeric problems, we equivalently compute

$$\log p_B(A) = \sum_{i=1}^{S_A} \log p_B(\mathcal{F}_i^A) \quad (3.7)$$

which is called the *log-likelihood of A given B.* This can be made symmetrical to create a well-behaved distance measure:

$$d_1(A, B) = \frac{1}{2}(\log p_B(A) + \log p_A(B)) \tag{3.8}$$

The computation of d_1 needs accessing both the GMM models of A and B and their respective MFCC vectors. However, MFCC vectors for all songs are prohibitive to store in the context of large databases. A typical 3-minute song is represented by 7751 frames (taking -say- 2048 points with 50% overlap at 44100 Hz), which, given e.g. $N = 10$ MFCCs per frame, amounts to around 80,000 double precision floating-point numbers (640 Ko). Since the timbre distance is meant to integrate into a large scale meta-database architecture, we need to be able to compare the models themselves, without storing the MFCCs.

Kullback-Leibler divergence

A natural distance measure between two probability distributions p_A and p_B is given by the *Kullback-Leibler divergence* (KL), also called *relative entropy*, or *information divergence*. It can be interpreted in information theory as the expected extra message-length per datum that must be communicated if a code that is optimal for a given (wrong) distribution Q is used, compared to using a code based on the true distribution P.

$$d_{KL}(p_A||p_B)) = \int p_A(x) \log \frac{p_B(x)}{p_A(x)} dx \tag{3.9}$$

The KL-divergence has an analytical form in the case of simple gaussian distributions:

$$d_{KL}(\mathcal{N}_A||\mathcal{N}_B) = \frac{1}{4} tr(\Sigma_A \Sigma_B^{-1} - \Sigma_B \Sigma_A^{-1}) + (\mu_A - \mu_B)^T (\Sigma_B^{-1} - \Sigma_A^{-1})(\mu_A - \mu_B) \tag{3.10}$$

where *tr* is the “trace” operator. However, no such close form exists for mixtures of several Gaussian components.

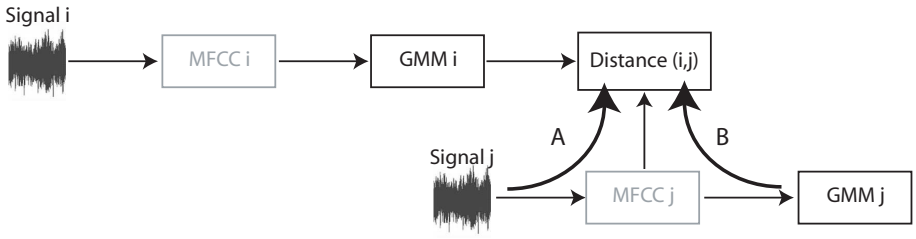

Figure 3.5: Comparison of the Log-likelihood approach and the MonteCarlo approach for the computation of the Kullback-Leibler divergence between 2 GMMs i and j. The MFCC vectors of each song are too space-consuming to be stored. Hence, there are 2 possible routes to compute the probability of the features of song j given the model of song i. Route A consists in temporarily recomputing the MFCCs from the signal j for the sake of the distance computation. Route B consists in sampling "fake" MFCC points from the already computed GMM of j, which is naturally much quicker.

We therefore propose to compute Equation 3.9 by Monte-Carlo approximation (Fishman, 1996), i.e. to estimate the KL distance by the empirical mean :

$$\widetilde{d(A,B)} = \frac{1}{n}\sum_{i=1}^{n} \log \frac{p_B(x_i)}{p_A(x_i)} \tag{3.11}$$

(where n is the number of samples x_i drawn according to p_A). The estimate $\widetilde{d(A,B)}$ converges to $d_{KL}(A,B)$ when $n \to \infty$ by virtue of the central limit theorem :

$$\lim_{n\to\infty}(\frac{1}{n}\sum_{i=1}^{n} X_i - \mathcal{E}(X)) = \frac{1}{\sqrt{n}}\mathcal{N}(0,\sigma^2) \tag{3.12}$$

where X is the random variable $\log \frac{p_B(x)}{p_A(x)}$, X_i a realization of X, $\mathcal{E}(X)$ the mean of X and $\mathcal{N}(0,\sigma^2)$ a normal distribution of mean 0 and variance σ^2 equal to the variance of X.

More precisely, we sample a large number of points $\mathcal{S}^{\mathcal{A}}$ from model A, and compute the likeli-

hood of these samples given model *B*. We then make the measure symmetric and normalize :

$$\begin{aligned} D(\mathcal{A},\mathcal{B}) \quad = \quad & \sum_{i=1}^{i=\mathcal{DSR}} \log \mathcal{P}(\mathcal{S}_i^{\mathcal{A}}/\mathcal{A}) + \sum_{i=1}^{i=\mathcal{DSR}} \log \mathcal{P}(\mathcal{S}_i^{\mathcal{B}}/\mathcal{B}) \\ & - \sum_{i=1}^{i=\mathcal{DSR}} \log \mathcal{P}(\mathcal{S}_i^{\mathcal{A}}/\mathcal{B}) - \sum_{i=1}^{i=\mathcal{DSR}} \log \mathcal{P}(\mathcal{S}_i^{\mathcal{B}}/\mathcal{A}) \end{aligned} \tag{3.13}$$

The precision of the approximation is clearly dependent on the number of samples, which we call Distance Sample Rate (DSR).

The Monte Carlo approach can be seen as a way to "recreate" temporary, prototypical MFCCs points from the GMMs, instead of keeping the original MFCCs from the audio signals (which can't be stored realistically). Sampling points from a GMM is naturally much quicker than re-processing the whole MFCC algorithm on the original signal. Figure 3.5 illustrates a comparison of the 2 approaches.

Sampling from GMMs

The Monte-Carlo approximation of the KL divergence requires to sample points from Gaussian Mixture Model distributions. To do so:

1. First draw a gaussian component at random among the *M* components of the GMM, according to the w_m weights (which sum to one $\sum_{m=1}^{M} w_m = 1$, and therefore can be considered as a discrete probability distribution on the space of components).

2. Then draw a point from the selected gaussian component.

An efficient way to draw points from an individual gaussian distribution uses a source of uniform pseudo-random numbers, and a mathematic transformation of these numbers such that their resulting distribution is gaussian. The most basic of such transformations is called the *Box-Muller*

transform (Press et al., 1986) :

$$y_1 = \sqrt{-2 \ln x_1} \cos 2\pi x_2 \tag{3.14}$$

$$y_2 = \sqrt{-2 \ln x_1} \sin 2\pi x_2 \tag{3.15}$$

Hence,

1. Generate two independent random numbers, x_1 and x_2 with a uniform distribution (in the range from 0 to 1), using e.g. the high-quality *Mersenne Twister* pseudo random number generator (Matsumoto and Nishimura, 1998).

2. Then apply the above transformations to get two new independent random numbers which have a Gaussian distribution with zero mean and a standard deviation of one.

3. Then rescale y_1 and y_2 to the right mean and variance, using $y' = \sigma(y + \mu)$

In practice, for numerical stability reasons, the polar form of the Box-Muller transform is preferred:

$$y_1 = x_1 \sqrt{\frac{-2 \ln R}{R}} \tag{3.16}$$

$$y_2 = x_2 \sqrt{\frac{-2 \ln R}{R}} \tag{3.17}$$

where $R = x_1^2 + x_2^2$.

3.2 MIR Design Patterns and Heuristics

The algorithm described above is designed as an attempt to explicitly model polyphonic timbre similarity, in order to test the underlying assumptions of a large class of automatic music description systems. Only a few previous attempts at building audio similarity functions can be found in the literature. Foote (1997) presents a system that also uses cepstral coefficients as a front-end, but rather uses a supervised algorithm (tree-based vector quantizer) that learns the most distinctive dimensions

in a given corpus. Adding one song to this corpus requires to redo the learning of the tree, which is expensive. On the contrary, our system is completely scalable, since it models each song separately.

Welsh et al. (1999) proposes a query by similarity system that is also able to match songs according to their timbre. He uses a large set of features (1248 floating-point per song) which are compared with the euclidean distance. However, his system doesnt address timbre similarity explicitly: his features model the pitch/tonal content of a song ("returning songs in the same key"), the noise level ("whether it is pure classical music or noisy, saturated hard rock") and the rhythm. The timbral similarity observed in some results by the author ("a pop, male vocal song produces results where every song in the top 10 is a male vocal with guitar and drum accompaniment") appears therefore as a side-effect of the features above, notably those describing the tonal content of the pieces. Our system is both more restrictive, and more precise: notably, the features that we use are meant to be independent of the pitch. We do not try to model music similarity at large, but only timbral similarity, which is is only one similarity relationship among many others (rhythm, melody, style, structure, etc.), some of which addressed by Welsh. We have argued in Aucouturier and Pachet (2002b) that the interestingness of a music retrieval system probably lies in the confrontation between several such similarity relationships.

Finally, Logan and Salomon (2001) proposes a similar approach to ours, which also uses Cepstrum Coefficients, only with a different modelling and a more complex matching algorithm. It is only since this contribution and our original formulation of the above mentioned algorithm in Aucouturier and Pachet (2002b) that "timbre similarity" has seen a growing interest in the Music Information Retrieval community (Baumann (2003); Baumann and Pohle (2003); Berenzweig et al. (2003); Herre et al. (2003); Kulesh et al. (2003); Pampalk et al. (2003); Pampalk (2004); Flexer et al. (2005); Stenzel and Kamps (2005); Vignoli and Pauws (2005)).

A careful inspection of the methodology of each of these contributions as well as the very many papers implicitly relying on the same model reveals a number of transformations that are commonly applied to the prototypical algorithm, and a number of heuristics used to guide research and algorithm design. We attempt here a catalog of such *design patterns*, in the spirit of Gamma

et al. (1995). It is often striking that the same patterns have also proved useful in the longer history of Automatic Speech Recognition (ASR) research, which we will highlight when relevant.

3.2.1 Pattern: Tuning feature parameters

Definition

Finding feature parameter values that improve the quality of the whole algorithm.

Description

A *feature* is a mathematical transformation of the input musical signal, aiming at reducing its dimensionality and variability. Good features typically are a measurement of a reduced aspect of the data which is believed to be relevant to the problem being modelled, or similarly a way to discard information which is not relevant. In the case of the modelling of musical timbre, following the insights from the psychoacoustical studies described above, we may want to only focus on spectral information, but also to discard fine harmonic details in the spectrum that typically vary a great deal with the pitch of the analysed sounds. The distinction between such feature extraction and the modelling of the extracted features (which may also discard e.g. dimensions in the feature space) is not always clear cut. Often feature extraction can be regarded as a fixed transformation of the input data, whereas the model itself contains adaptive parameters whose values are set as part of a training process (Bishop, 1995). However, the statistical model one uses is crucially dependent on the choice of features. Hence it is often useful to fine-tune some parameters of the feature extraction algorithm, to adapt e.g. its accuracy.

A typical parameter in feature extraction influences the dimension of the measurement made on the data, and thus the dimension of the space (the *feature space*) in which statistical modelling is performed. For instance, the number of MFCC coefficients extracted from each frame directly controls the dimension of the feature vectors. Increasing the dimension of the features typically increases the expressiveness of the corresponding representation, thus allowing

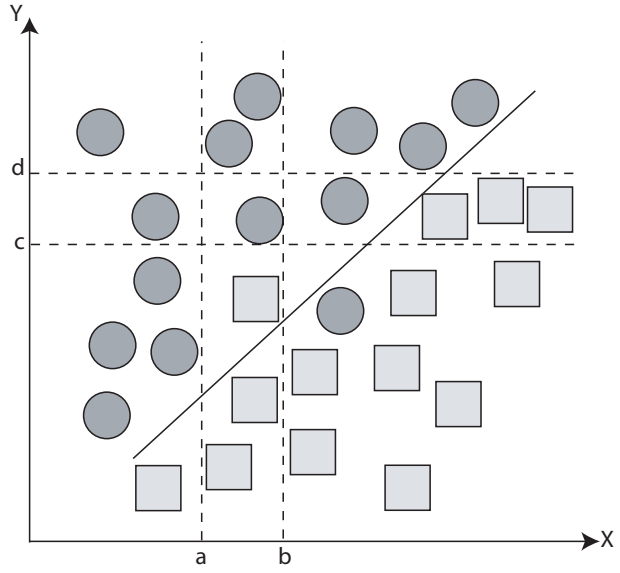

Figure 3.6: Illustration of the influence of feature dimension on a 2 class classification problem.

e.g. to better discriminate different classes of data. Figure 3.6 illustrates a possible 2-class classification problem, using 2 abstract, one-dimensional features X and Y. Data points of the 2 classes are represented by dark gray circles and light-gray squares respectively. It appears that each of the features X and Y is a poor representation for the problem when considered individually. For instance, the observation of the value of $X \in [a, b]$ for a point gives no information about its most likely class: 3 examples of each class are observed within this range. Similarly, the observation of $Y \in [c, d]$ is non informative. However, when considered jointly, the observation of $(X, Y) \in [a, b] \times [c, d]$ gives a good indication that the class of the corresponding point be "circles". This suggests that increasing the number of features is a general law for improving the precision of the modelling. However, we often find that, beyond a certain point, adding new feature dimensions can actually lead to a reduction in the precision of the modelling. This phenomenon is known as the *curse of dimensionality*. It can be understood on the same Figure 3.6. The more dimensions d are available in the feature space, the more precise a partition of the data space we can obtain. However, the number of cells such as $[a, b] \times [c, d]$ grows exponentially with d. If we keep the number of training

Table 3.1: Influence of the number of MFCCs on a similarity task, reproduced from Logan and Salomon (2001).

Number of MFCC	Precision5	Precision10
12	3.43	6.53
19	3.44	6.57
29	3.36	6.44

data points constant, and increase d, there will come a point where no data will be available for most cells, and the corresponding representation will be very poor. In Figure 3.6, where $d = 2$, only one data example is found in $[a, b] \times [c, d]$. If we increment d, most cells in $[a, b] \times [c, d] \times R$ will be empty, and the precision of the model will collapse.

Example

In Logan and Salomon (2001), the authors test the influence of 3 numbers of MFCCs used as front end for a timbre similarity measure (later modelling the MFCC distributions with the kMeans method, and comparing the distributions with Earth Mover's Distance). Table 3.1 reproduces their results. In the table, the precisions are measured as the average number of songs with the same genre as the seed song in the first 5 nearest neighbors of the seed ("Precision5") and in the first 10 nearest neighbors ("Precision10"). We observe with the authors that increasing the dimensionality of the MFCCs from 12 to 19 has a small positive impact on the precision of the model, but that further increase degrades the precision, as explained by the curse of dimensionality.

3.2.2 Pattern: Tuning model parameters

Definition

Finding values for parameters of the statistical model that improve the quality of the whole algorithm.

Description

As for features (3.2.1), statistical models used to learn the distribution of the features often

have a number of fixed parameters that have to be chosen before training. These parameters typically influence the complexity of the model, i.e. its expressiveness. For instance, the number of Gaussian components M in a Gaussian Mixture Model (GMM) influences the flexibility of the density estimation that it is able to achieve. It can be theoretically proved (Bishop, 1995) that a GMM is a universal probability density estimator if M is allowed to grow to infinity. However, as above with features, increasing the complexity of a model does not necessarily improves the precision of the corresponding representation of the systematic aspect of the data.

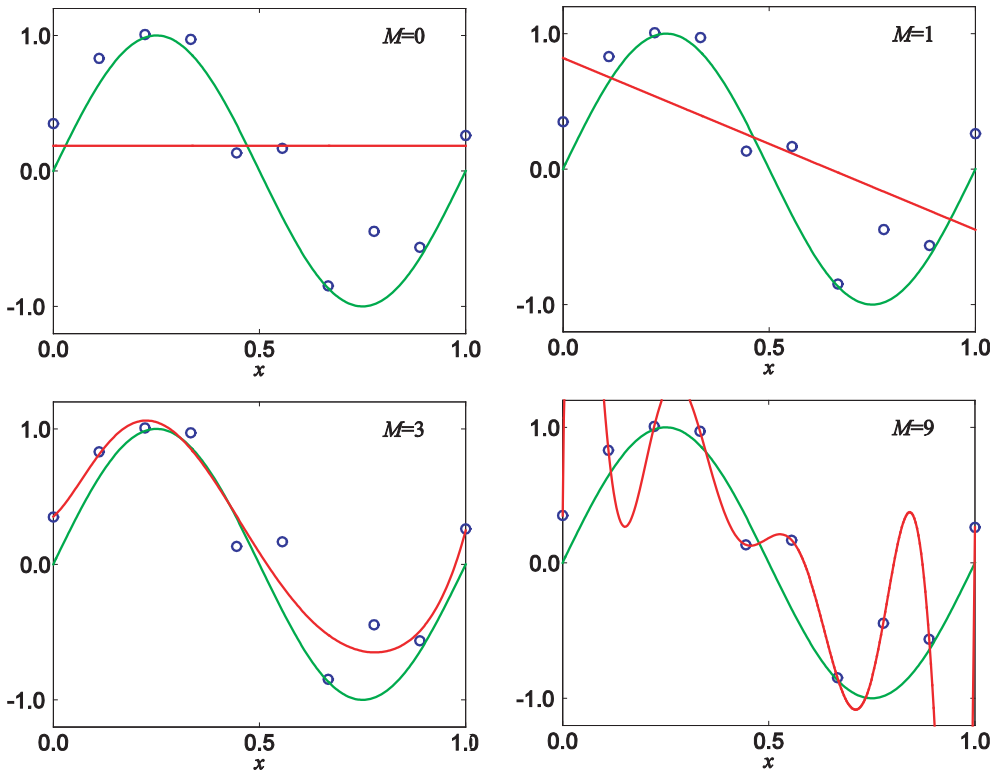

Figure 3.7: Illustration of the influence of model complexity on a polynomial regression task. Pictures reproduced from Bishop (1995)

This is best illustrated in the case of a simple polynomial curve fitting, as shown in Figure

3.7. We generate training data from the function

$$f(x) = 0.5 + 0.4sin(2\pi x) \tag{3.18}$$

and add a small amount of noise. We then try to model the data with a Mth-order polynomial given by

$$y(x) = \sum_{i=0}^{M} w_i x^i \tag{3.19}$$

Figure 3.7 shows the least-square error estimates of the polynomial for 4 values of the parameter $M \in \{0, 1, 3, 9\}$, which controls the complexity/flexibility of the model. Models with $M = 0$ and $M = 1$ give a poor representation of $f(x)$, due to their limited flexibility. $M = 3$ shows a much better fit to the data, and thus a better approximation of the underlying process $f(x)$. However, when we further increase $M = 9$, the approximation of the underlying function actually gets worse, although we achieve a perfect fit to the training data. The added flexibility of the model is spent on modelling non-meaningful noisy variations on the data, thus degrading the generalisability of the model to non-previously seen data. Such models are said to be *over-fitted* to the data. Therefore, the best performance is typically achieved by a model whose complexity is neither too small or too large.

Examples

In Nwe and Wang (2004), the authors compare the performance of a vocal segment detector using 12-dim MFCCs and a hidden Markov model (HMM), for 2 and 10 gaussian mixtures per state of the HMM, and report and average precision of 81.3% and 74.6% respectively. The poorer performance of the most flexible model is a clear case of overfitting. In Berenzweig et al. (2003), the authors compare the precision of a timbre similarity task for different numbers of clusters in a k-mean model in a 20-dim MFCC space. The metric used to estimate the precision of the similarity is the average number of queries in a test database for which the first nearest neighbor is the same as a ground truth similarity measure (“first place agree-

Table 3.2: Influence of the number of clusters of KMean modelling on the precision of a similarity task, reproduced from Berenzweig et al. (2003).

Number of clusters	First place agreement percentage
8	21%
16	22%
32	23%
64	23%

ment"). The reported results, which we reproduce in Table 3.2, show little influence of the parameter over the tested range of values.

3.2.3 Pattern: Feature Equivalence

Definition

Replacing a feature by an "equivalent" feature.

Description

Features as they are introduced by various researchers tend to gather in sets of equivalent semantics. *Temporal features* especially contain information about the duration of excitation of individual notes. From the short-time rms-energy envelope, one can estimate e.g. rise-time, decay-time, strength and frequency of amplitude modulation, crest factor and detected exponential decay from the rms-energy curve. As discussed above, such features have proved important in early psychoacoustic experiments in timbre perception, which focus on recordings of individual notes, however they tend to be difficult to use with full-length polyphonic music. *Spectral features* are considered more robust to polyphonic and complex textures. Most of them are based on the short-time Fourier transform (STFT), and have the general equivalent semantic of characterizing the global "spectral shape", in a compact and scalable form. We detail here two notable approaches to model the spectral shape, which lead to the definition of several equivalent sets of spectral features.

- Spectral Moments: The spectrum $F_t(w)$ of each frame s_t of signal is considered as a

probability distribution which observed values are the frequencies w and the probabilities to observe them are the normalized amplitudes $p(w) = \frac{F(w)}{\sum_w F(w)}$. From this distribution, one can compute the central moments. The motivation is that the series of the moments of the distribution can be used to approximate the distribution in a scalable way, i.e. the first moments contribute the most to the approximation, and the more moments, the more precise the approximation. This can notably be achieved (Cramér, 1957) with the Edgeworth series or Gram-Charlier A series, which if we include only the first two development terms, reads:

$$F(x) \simeq \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]\left[1 + \frac{\kappa_3}{\sigma^3} h_3\left(\frac{x-\mu}{\sigma}\right) + \frac{\kappa_4}{\sigma^4} h_4\left(\frac{x-\mu}{\sigma}\right)\right] \tag{3.20}$$

where $h3 = (x3 - 3x)/3!$ and $h4 = (xt - 6x2 + 3)/4!$ (Hermite polynomials), and κ_3 and κ_4 are the third and fourth cumulants of the distribution. κ_3 and κ_4 are related to the central moments m_3 and m_4 by:

$$m_3 = \kappa_3 \tag{3.21}$$

$$m_4 = \kappa_4 + 3\kappa_2^2 \tag{3.22}$$

The first moment of the spectrum, named *Spectral Centroid*, is the mean value of the frequency distribution, i.e. the barycenter of the spectrum.

$$\mu = \sum_w w p(w) \tag{3.23}$$

The second central moment, *Spectral Spread* is the spread of the spectrum its barycenter, i.e. the variance of the frequency distribution:

$$\sigma^2 = \sum_w (w-\mu)^2 p(w) \tag{3.24}$$

Similarly, we define the *Spectral Skewness* $\gamma_1 = \frac{m_3}{\sigma^3}$ and *Spectral Kurtosis* $\gamma_2 = \frac{m_4}{\sigma^4}$ using the third and fourth moments m_3 and m_4.

- Cepstrum: Another way to encode the shape of the spectrum in a scalable way is to consider the spectrum values $F_t(w)$ of each frame s_t as a signal of w, and to compute its Fourier transform, i.e. the Fourier transform of the Fourier transform. The infinite series of Fourier coefficients can be used to reconstruct the original signal, and as illustrated in Figure 3.8, the truncated series of the first coefficients can be used to approximate the signal in a scalable way. As already described in 3.1, the Fourier transform of the (decibel) spectrum is called *cepstrum* (an anagram of "spectrum", formed by reversing the first four letters). There exist many ways to compute the cepstrum, and many features built on the cepstrum, such as the notorious Mel-Frequency Cepstrum Coefficients. It is interesting to note that the MPEG-7 standard (ISO, 2002) formalizes such a set of transforms to describe the spectral shape, by defining *AudioSpectrumEnvelopeD* which is calculated by linear transformation of the STFT Power Spectrum

$$\mathbf{X} = |FT|^2_{\frac{N}{2}+1}\mathbf{T} \tag{3.25}$$

and *AudioSpectrumProjectionD*, which is calculated by converting *AudioSpectrumEnvelopeD* to a decibel scale and applying a decorrelating linear transform matrix $\mathbf{V}$

$$\mathbf{Y} = 10log_{10}(\mathbf{X})\mathbf{V} \tag{3.26}$$

This theoretically defines a class of equivalent features $\{\mathbf{Y}\}_{\mathbf{T},\mathbf{V}}$, based on the cepstrum, among which we find MFCC by choosing $\mathbf{T}$ to be the linear-to-Mel frequency map and $\mathbf{V}$ to be the discrete cosine transform (DCT).

Examples

Yang (2002) compares two alternative envelope features, standard MFCC and so-called

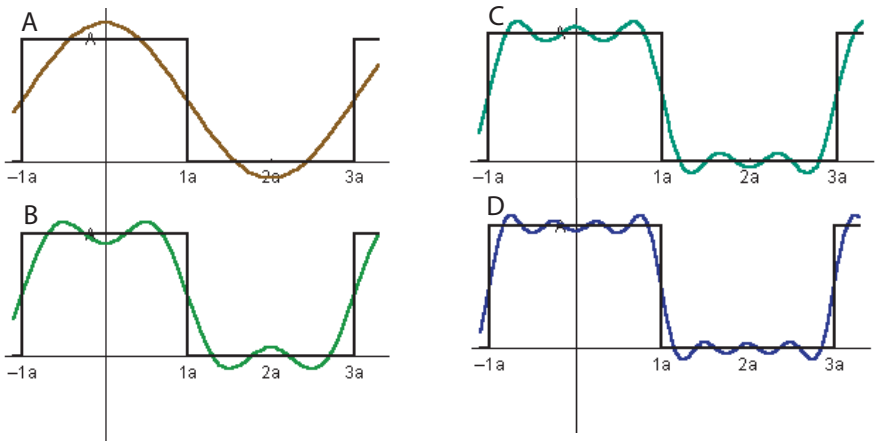

Figure 3.8: Reconstruction of a square signal using increasing numbers of its Fourier components (A:2, B:3, C:4, D:5).

"comb method" which convolves the spectrum with a family of comb filters. Casey and Crawford (2004) compares two different instantiation of the MPEG7 *AudioSpectrumProjectionD*, namely standard MFCC versus one built with an octave-based filterbank and a singular value decomposition. Tzanetakis and Cook (2000); Kapur et al. (2004); West and Cox (2004) compares a moment-like feature set (Spectral Centroid, Spectral Rolloff, Spectral Flux) to MFCCs, for various classification problems.

3.2.4 Pattern: Model Equivalence

Definition

Replacing the statistical model by an "equivalent" model

Description

As for features (3.2.3), equivalence classes of statistical models tend to emerge according to their high-level assumptions. A common partition considers "static" models such as Gaussian Mixture Models (GMM), histograms, k-Means or k-Nearest Neighbors (kNN), which estimate the probability distribution of the frames as a whole without taking any account of their time ordering on the one hand, and "dynamic" models on the other hand, such as hidden

Markov models (HMM) or Recurrent Neural Networks (R-NN), which model both the static distribution of the data and the time evolution of the distribution.

Examples

K-means are used in timbre similarity tasks by Logan and Salomon (2001); Baumann (2003); Berenzweig et al. (2003); Herrera-Boyer et al. (2003); Pampalk et al. (2003). GMMs are used in Berenzweig et al. (2003) and Kulesh et al. (2003). Scaringella and Zoia (2005) compares several dynamical models for a genre classification task, based on MFCC: R-NNs, Explicit-Time Modelling Neural Networks (Soltau, 1998), and HMMs.

3.2.5 Pattern: Feature Composition

Definition

Modifying a standard feature algorithm chain by inserting an additional mathematical operation.

Description

It is very common that an author should create a local variant of a standard feature by adding

- pre-processing such as low-pass filtering the signal, or normalizing its energy (e.g. $MFCC(Normalize(x))$)
- post-processing such as taking the derivative (so called delta-coefficients) or rescaling
- intra-composition i.e. changing or adding an operation block in the middle of the chain

Examples

In Yang (2002), MFCCs are computed not for each frame at a constant frame-rate, but only on the frames corresponding to a peak in the signal power ("event frames"). In Jiang et al. (2002), the authors propose to modify the MFCC algorithm to not only compute the average spectrum in each frequency band, but also a correlate of the variance, the *spectral contrast* (namely the amplitude between the spectral peaks and valleys in each subband). This modifies the

algorithm to output 2 coefficients (instead of one) for each Mel subband. Additionally, in the algorithm published in Jiang et al. (2002), the authors replace the Mel filterbank traditionally used in MFCC analysis by an octave-scale filterbank (C_0-C_1, C_1-C_2, etc.), which is assumed to be more suitable for music. They also decorrelate the spectral contrast coefficients using the optimal Karhunen-Loeve transform. The derivation of the standard MFCC algorithm to the SpectralContrast algorithm is illustrated in Figure 3.9.

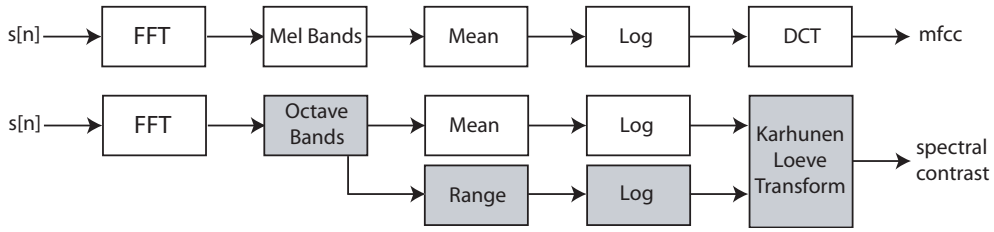

Figure 3.9: Comparison of the standard MFCC algorithm chain (upper diagram) with the Spectral Contrast algorithm (lower diagram), showing the various insertions / replacements.

3.2.6 Pattern: Cross-Fertilisation

Definition

Borrowing a technique, often a feature, which was developed and proved successful in pattern recognition for other domains than music.

Description

Music audio pattern recognition developed out of a large corpus of work done in the context of speech signals, for which a scientific and commercial interest was recognized earlier, notably on the impulsion of the 1971 DARPA call on Speech Understanding Research (SUR) (Kurzweil, 1996). The research effort of the five-year SUR project, which targeted a non real-time recognition system with 90% sentence accuracy for continuous-speech sentences using thousand word vocabularies, notably lead to such great advances as dynamic programming (Itakura, 1975), and Markov modelling (Itakura, 1976). Cepstrum (see 3.2.3) has been the dominant feature for speech recognition, notably since the classic formulation of Mel-

Frequency Cepstrum Coefficients (MFCCs) by Rabiner (1989). However, it was originally invented for characterizing the seismic echoes resulting from earthquakes and bomb explosions (Tukey and Healy, 1963). Their success in the speech community logically lead to their application to music signals, which can be traced back to the best of our knowledge to Foote (1997), inspired by previous work on Speaker Recognition (Foote and Silverman, 1994). Logan and Salomon (2001), also coming from Speech Recognition, give a detailed account of how well the assumptions of MFCCs hold for musical signals. Since then, it is a common pattern that features and techniques for timbre and music modelling should be borrowed from other domains, not only speech recognition, but also sismic data or image processing.

Examples

Kim and Whitman (2002) uses Linear Predictive Coding (LPC) (Rabiner and Juang, 1993) , a technique used for Speech Compression, and a variant thereof, to build a singer identification system. LPC are functionally equivalent to MFCC, as they encode the spectral envelope of the signals, however they are believed to be better suited to the sharp formant spectrum exhibited by voice signals, due to their formulation as a all-pole filter function modelling the resonances of the vocal tract. Unfortunately, no comparison is given with standard MFCCs. Deshpande et al. (2001) proposes to consider audio spectrograms as images, and to extract features developed in the context of image texture classification (convolution with directional gaussian filters). Similarly, Casagrande et al. (2005) uses Haar-filters, a technique borrowed to object image detection, to detect regular geometric patterns in speech and non-speech audiospectrograms.

3.2.7 Pattern: Modelling Dynamics

Definition

Modifying the features and/or the model to account for the dynamics of the data.

Description

The prototypical algorithm described in 3.1 does not take any account of time scale greater than the frame size. All frames are typically modelled as a whole, without any account of their time ordering. It is a common improvement strategy to many contributions to modify this prototypical algorithm so as to take the dynamics of the data into account. Modifications may occur at the feature level, by e.g.

- tapped delay line: consecutive feature vectors can be stacked into n-times larger feature vectors, before sending them to the statistical model, thus constructing a flat spatial embedding of the temporal sequence (see Figure 3.10).

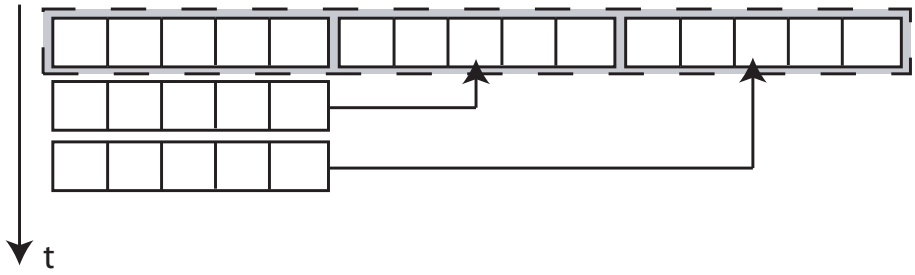

Figure 3.10: Tapped delay line to account for the dynamics of the static features.

- derivation: Furui (1986) showed that speech recognition performance can be greatly improved by adding time derivatives to the basic static features. Delta Coefficients are computed using the following formula:

$$d_t = \frac{\sum_{\theta=1}^{\Theta} \theta(f_{t+\theta} - f_{t-\theta})}{2\sum_{\theta=1}^{\Theta} \theta^2} \tag{3.27}$$

  where d_t is a delta coefficient at time t, computed using a time window Θ. The same formula can be applied to the delta coefficients to obtain the acceleration coefficients. The resulting modified feature set contains, for each frame, the static feature values and their local delta values (see Figure 3.11).

- texture windows: local static features (typically extracted every 50 ms) can be averaged over larger-scale windows (typically several seconds), in an attempt to capture the long-

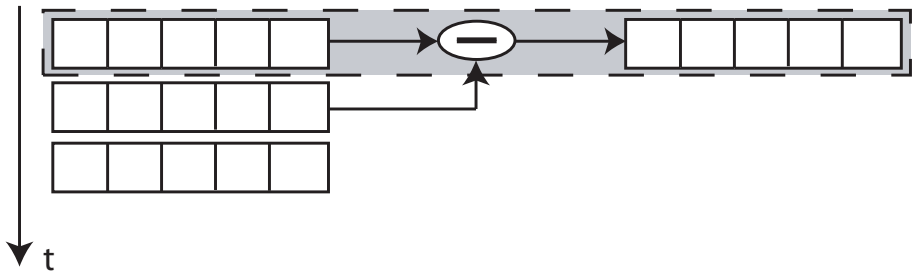

Figure 3.11: Computation of delta coefficients from a sequence of static features.

term nature of sound textures, while still assuring that the features be computed on small stationary windows. Several statistics can be used on such so-called *texture windows*, e.g. mean, standard deviation, skewness, range, etc. (see Figure 3.12).

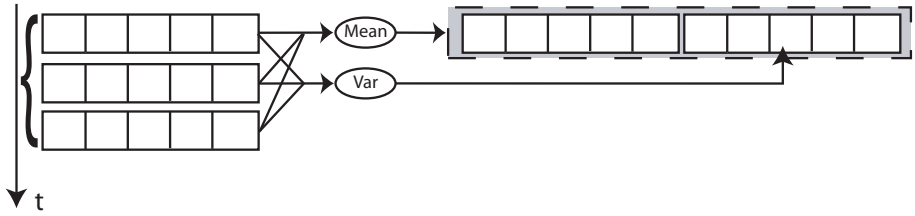

Figure 3.12: Computation of statistics over texture windows.

- dynamic features: Another strategy to characterize the dynamics of the static features is to compute features on the signal constituted by the static feature sequence (which is considered to be sampled at the original frame-rate). For instance, a high-resolution STFT can be taken on large frames (of several seconds' duration), and the low-frequency variations of the features (e.g. $[1-50Hz]$) are taken as features instead of the original ones (see Figure 3.13).

Dynamic models can also be used to account for the dynamics of static features. Such models include recurrent neural networks (R-NN), which are typically 2 layer networks with feedback from the first layer output to the first layer input, and hidden Markov models, which can be defined as a set of GMMs (also called states) linked to one another by a matrix of transition

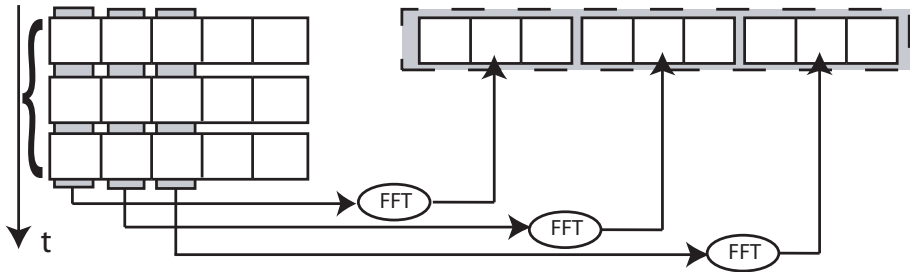

Figure 3.13: Computation of dynamic features with FFT over the sequence of static features.

probabilities.

Examples

Use of Delta coefficients for MFCCs feature sets is reported e.g. in Talupur et al. (2000); Tzanetakis and Cook (2000); Heittola and Klapuri (2002). Tzanetakis and Cook (2002) reports a convincing 15% precision improvement on a genre classification task when using MFCC texture windows of up to about 40 frames (1 second). Whitman and Ellis (2004) uses a modulation spectrum constructed by taking the long-term FFT of the MFCC, and aggregating the obtained spectrum for each cepstral coefficient in 6 octave wide bins in $[0, 50Hz]$, while Peeters et al. (2002) directly takes the FFT of the Mel-filtered spectrum, and keeps only the spectrum values that maximize the mutual information. Scaringella and Zoia (2005) compares the performances of several static models (support vector machines) with different dynamic feature schemes, such as texture windows and delay lines, on a genre classification task. The authors notably conclude that a 1 second delay line performs better (69%) than 1 second texture windows (65%).

3.2.8 Pattern: Higher-level knowledge

Definition

Improving the models by adding a layer of higher-level, non-signal intelligence.

Description

This pattern can also be observed when reviewing Automatic Speech Recognition (ASR) research. Over the last 30 years, much effort has been done to refine the signal representation of phoneme utterances, by the introduction of psychoacoustic models, and the statistical modelling of the phoneme dynamics, notably using markovian probabilistic frameworks. However, usable ASR performances could only be reached by combining such low-level techniques with higher-level Natural Language Processing (NLP) architectures such as sentence parsers and grammar models. Such systems can e.g. be used to disambiguate word boundaries using contextual information and/or common sense. The point was notoriously demonstrated by one famous T-shirt worn by Apple Computer researchers, reading: "I helped Apple wreck a nice beach", which, when spoken, sounds like "I helped Apple recognize speech". A review of NLP techniques and their integration with Speech Recognition can be found in Cole et al. (1996).

Examples

Nwe and Wang (2004) describe a system to take song structure information into account when classifying vocal/non vocal music. Their observation is that vocal and non vocal signal segments display intra-song variations, such as signal energy, depending on the "song section": intro, verse, chorus, bridge and outro. Hence, they manually split the training data into several sub-classes corresponding to each section. Using this scheme, they report a small improvement (2%) over standard approaches. McKay and Fujinaga (2004) propose to use a hierarchical genre taxonomy to improve automatic genre classification. Models are built both for broad genre categories, such as Classical or Jazz ("root nodes"), and for more precise sub-categories such as Jazz/Bebop or Jazz/Swing ("leaf nodes"). As can be seen in Table 3.3, prefiltering recordings by root genre, before proceeding to the next level of the hierarchy with the subcategories of the winner root moderately improves the precision over the standard flat approach in which all categories, root and leaf, are considered at the same stage.

Table 3.3: Influence of using hierarchical taxonomies in genre classification. Results reproduced from McKay and Fujinaga (2004).

Classification type	Precision on root genres	Precision on leaf genres
Flat	96%	86%
Hierarchical	98%	90%

3.3 Conclusion

The patterns described in this Chapter translate our personal view of the research made in the field of Music Information Retrieval. The main objective of this description is rhetorical, as it provides a way to structure the exploration made in the next chapter of the algorithmic space of typical models of timbre. All these variations elaborate on the same common assumption, namely that the perception of polyphonic timbre results from the long-term distribution of local, spectral features. It is the validity of this assumption that we test in the next Chapter (**Experiment 1**).

However, we discuss here the possibility of other applications of such a structured view on the domain. First, MIR patterns have pedagogical value*. They provide a condensed and transverse view on the enormous body of literature that emerged in the last 5 years or so. They help linking contributions to one another, by identifying common methodologies. However, these patterns don't truly constitute an analysis method for MIR literature, insofar as they don't yield any conclusion other than the fact that there *are* patterns.

The patterns described here go some way in the direction of formalizing research creativity and intuitions. Notably, they support metaphorical reasoning, e.g. "if delta coefficients work to some extent, I should probably investigate texture windows or hidden Markov models" (pattern 3.2.7). However, they're not a truly operational framework, as are design patterns in software engineering. Notably, they don't provide prototypical solutions to given problems. Thus they cannot easily be used to help or even automatize research. However, some of the heuristics described here are in the line of the automatized approach of the EDS system (Zils and Pachet, 2004). EDS proposes to discover new signal-processing features by composition of a set of basic operators, guided by

*Actually they were used by the author in a course given at ENSEIRB, Bordeaux, France in December 2005, on the topic of music pattern recognition

genetic programming.

Such patterns could be seen as a cynical view on the field, carrying the condescending message of a lack of variety and originality. However, the existence of such a - rather small in fact - number of patterns in "Signal Software Engineering" is probably little more than a logical consequence of the dominant paradigm of pattern recognition: this pretty much imposes a finite number of variations around the theme of features and models. This in particular seems a less flexible paradigm than the Object Oriented approach presiding over design patterns in Software Engineering.

In fact, these patterns tell more about the scientific discourse than the science itself. They provide a genealogy of prototypical concepts and methods, and could e.g. be used to help generate "state of art" sections for new papers, or to organize bibliographical collections of papers, such as the ISMIR repository[†].

On the whole, the patterns described in this chapter ask the question of the knowledge representation - not of the world under scientific inquiry - but of the domain-specific scientific method itself. In particular, we would like to see systems that can tap automatically into existing contributions (papers, code, databases, etc.) to help the research on new problems, by suggesting features or methods. The field of Music Information Retrieval with its infinity of specific descriptor problems (new genres, new instruments, new moods, etc.), and its rather constrained and clustered space of solutions (cf. the patterns above), appears to be a possible candidate for such systems.

[†]http://www.ismir.net/all-papers.html

Part II

Experiments

Chapter 4

Experiment 1: The Glass Ceiling

In this chapter, we test the validity of the assumptions underlying the pattern recognition approach to polyphonic timbre, by hill-climbing the algorithmic space described in Chapter 3. In particular, we question the common assumption found in the literature that error rates reported on implicit models of timbre are incidental, and that near-perfect results would just extrapolate by fine-tuning the algorithms parameters. We describe an evaluation framework which targets explicitly the notion of timbre similarity, instead of derived high-level descriptions. We then use this framework to evaluate the precision of very many parameters and algorithmic variants, some of which have already been envisioned in the literature, some others being inspired from typical patterns of research methodologies observed earlier.

This leads to an absolute improvement over existing algorithms of about 15% precision. But most importantly, we describe many variants that surprisingly do not lead to any substancial improvement of the measure's precision. Moreover, our simulations suggest the existence of a *glass ceiling* at precision about 65% which probably cannot be overcome by pursuing such variations on the same theme.

4.1 Experiment

We explore the space of polyphonic timbre models by applying transformations (so-called *design patterns*) to the prototypical algorithm described in Chapter 3.1. We list here all the variants that were tested. For a complete description of each variant, please refer to Appendix B.

Parameters of the original algorithm (patterns 3.2.1 and 3.2.2)

We explore the space constituted by the following parameters :

- The sample rate of the music signal (11, 22 or 44kHz).
- The size of the frames on which we compute the MFCCs (from 10ms to 1s).
- The number of the MFCCs extracted from each frame of data (from 10 to 50).
- The number of gaussian components used in the GMM to model the MFCCs (from 10 to 100).
- The number of samples used in Monte-Carlo approximation (from 1 to 10,000).

Replacing MFCC by MPEG-7 Spectral Descriptors (pattern 3.2.3)

We try replacing/appending to the MFCC feature set various combinations of MPEG7-standardized spectral descriptors based on moments of the spectrum, as described in Section 3.2.3: `SpectralCentroid`, `SpectralSpread`, `SpectralKurtosis`, `SpectralSkewness`, `SpectralFlatness`, `SpectralRolloff`, `SpectralFlux`.

Alternative distance measure (pattern 3.2.4)

Some authors (Logan and Salomon (2001); Berenzweig et al. (2003)) propose to compare the GMMs using the Earth Mover's distance (EMD), a distance measure meant to compare histograms with disparate bins (Rubner et al. (1998)). We evaluate 2 different implementations of EMD against the Monte Carlo approximation to KL divergence.

Non-Parametric Modelling (pattern 3.2.4)

We evaluate and fine-tune the parameters of 2 alternative models of the MFCC distribution than the baseline GMM algorithm:

- Pampalk's Spectrum histograms: 2D histogram counting the number of times loudness levels (out of 10 normalized values) are exceeded in a each frequency band (on 20 Bark bands) (Pampalk, 2004)
- MFCCs Histograms: based on vector-quantization of the MFCC space (Kohonen, 1995).

Processing commonly used in Speech Recognition (pattern 3.2.5)

MFCCs are a very common front-end used in the Speech Recognition community (Rabiner and Juang, 1993), and a variety of pre and post-processing has been tried and evaluated for speech applications. We examine the influence of 6 common operations:

- ZMeanSource: used in speech to remove the effects of A-D conversion.
- Pre-emphasis: used in speech to reduce the effects of the glottal pulses and radiation impedance and to focus on the spectral properties of the vocal tract.
- Dither: adding a small amount of noise to the signal to avoid numerical problems due to certain kind of waveform data (*finite wordlength effects*).
- Liftering: used in speech to rescale the MFCC coefficients to have similar magnitude.
- Cepstral mean compensation (CMC): to subtract the effect of the transmission channel.
- 0'th order coefficient: correlated with the signal's log energy

Alternative MFCC algorithm (pattern 3.2.5)

In Jiang et al. (2002), the authors propose a simple extension of the MFCC algorithm to not only compute the average spectrum in each band (or rather the spectral peak), but also a correlate of the

variance, the *spectral contrast* (namely the amplitude between the spectral peaks and valleys in each subband). We evaluate two different implementation of this variant.

Borrowing from Image Texture Analysis (pattern 3.2.6)

We evaluate techniques inspired from those used for the automatic analysis of image *textures*, notably second-order statistics analysis with grey-level co-occurrence matrix (GLCM) as proposed by Haralick et al. (1973).

Delta and Acceleration Coefficients (pattern 3.2.7)

It is known since Furui (1986) that the performance of a speech recognition system can be greatly enhanced by adding time derivatives to the basic static parameters. We evaluate the performance of adding delta and/or acceleration coefficients to the original MFCC dataset, with different parameter values.

Texture windows (pattern 3.2.7)

Tzanetakis in Tzanetakis and Cook (2002) reports that using a larger scale texture window and computing the means and variances of the features over that window results in significant improvements in music classification. We evaluate the influence of this variant, for window sizes between 0 and 100 frames.

Hidden Markov models (pattern 3.2.7)

To explicitly model this short-term dynamical behavior of the data, we try replacing the GMMs by hidden Markov models (HMMs, see Rabiner (1989)), and to fine tune the models' parameters (number of gaussian components, number of states).

Building in knowledge about note structure (pattern 3.2.8)

We investigate here 2 techniques to build in higher-level knowledge about the structure of musical notes, namely the segmentation between transient and steady state:

- Removing noisy frames: identified with Spectral Flatness (Johnston (1988))
- Note Segmentation: In typical implementations, MFCCs are computed with a constant frame-rate, and thus may average transient and steady-states of musical notes. We investigate whether synchronizing the MFCC extraction to a preliminary automatic note segmentation can improve the global modelling of polyphonic timbre similarity.

4.2 Method

4.2.1 Explicit modelling

We conduct our evaluation on an *explicit* model of the timbre similarity of polyphonic textures. By focusing on the low-level perceptive mechanism of timbre similarity, we aim at studying the validity of the approach shared by the contributions described above, without depending on the unknown correlations that exist at the level of high-level music descriptions.

4.2.2 Ground Truth

For this study, we have constructed a test database of 350 song items, as an extract from the Cuidado database (Pachet et al., 2004) which currently has 15,460 mp3 files. It is composed of 37 clusters of songs by the same artist, which were refined by hand to satisfy 3 additional criteria:

- First, clusters are chosen so they are as distant as possible from one another.
- Second, artists and songs are chosen in order to have clusters that are “timbrally” consistent (all songs in each cluster sound the same).

- Finally, we only select songs that are timbrally homogeneous, i.e. there is no big texture change within each song.

The test database is constructed so that nearest neighbors of a given song should optimally belong to the same cluster as the seed song. Details on the design and contents of the database can be found in Appendix A.

4.2.3 Evaluation Metric

We measure the quality of the measure by counting the number of nearest neighbors belonging to the same cluster as the seed song, for each song. More precisely, for a given query on a song $\mathcal{S}_i$ belonging to a cluster $C_{\mathcal{S}_i}$ of size $\mathcal{N}_i$, the precision is given by :

$$p(\mathcal{S}_i) = \frac{card(\mathcal{S}_k / C_{\mathcal{S}_k} = C_{\mathcal{S}_i} and \mathcal{R}(\mathcal{S}_k) \leq \mathcal{N}_i)}{\mathcal{N}_i} \tag{4.1}$$

where $\mathcal{R}(\mathcal{S}_k)$ is the rank of song $\mathcal{S}_k$ in the query on song $\mathcal{S}_i$.

The value we compute is referred to as the *R*-precision, and has been standardized within the Text Retrieval Community (Voorhes and Harman, 1999). It is in fact the precision measured after *R* documents have been retrieved, where *R* is the number of relevant documents. To give a global *R*-precision score for a given model, we average the R-precision over all queries (i.e. 350, which is the number of songs in the test database).

4.3 Tools

On the whole, the different variants and parameters examined in the experiment represent more than 500 algorithms. Such extensive testing over large, dependent parameter spaces is both difficult and costly. We describe here our efforts in making this study possible, both through software architectures, implementations and algorithmic innovations.

4.3.1 Architecture

Conducting the systematic evaluation of music similarity measures requires the building of a general architecture which is nearly as complex as a full-fledge EMD system, that is able to e.g. access and manage the collection of music signals the measures should be tested on, store each result for each song, compare results to a ground truth etc.

In the context of the European projects Cuidado and SemanticHifi, the music team at SONY CSL Paris has developed a fully-fledged EMD system, the Music Browser (Pachet et al., 2004) based on the MCM Java API, which is to our knowledge the first system able to handle the whole chain of EMD from metadata extraction to exploitation by queries, playlists, etc. Metadata about songs and artists are stored in a database, and similarities can be computed on-the-fly or pre-computed into similarity tables. Its open architecture makes it easy to import and compute new similarity measures. Similarity measures themselves are objects stored in the database, for which we can describe the executables that need to be called, as well as the arguments of these executables.

Figure 4.1 shows a screenshot of the Music Browser in a typical experimentation session. The "Metadata" panel (in the background) shows the list of all metadata available for items of the "song" type. Metadata appearing in light gray indicate unary descriptors of songs (which have a single value for a given song). Typical unary descriptors are timbre models (e.g. `CentroidFlatnessRolloffGLCM` which stores the Grey-Level Cooccurrence matrix of a given song computed from the concatenation of its Spectral Centroid, Flatness and Rolloff, see Section 4.1 and Appendix B.8) and various metrics used for the analysis of experimental results (e.g. `Angle GMM` measures the mean Neighbor angle for a given song, which is a measure we use in Chapter 6 to evaluate the hubness of a song). Metadata appearing in dark gray indicate similarity measures, which have a single value per duplet of songs. The highlighted distance in Figure 4.1, `Distance Histogram MFCC_KMEAN`, is a euclidean distance computed between histograms of MFCCs quantized with a KMean algorithm (see Section 4.1 and Appendix B.7.2). The parameters of the selected metadata can be edited in the right area of the Metadata panel. Additional metadata can be created from this same panel with a few clicks.

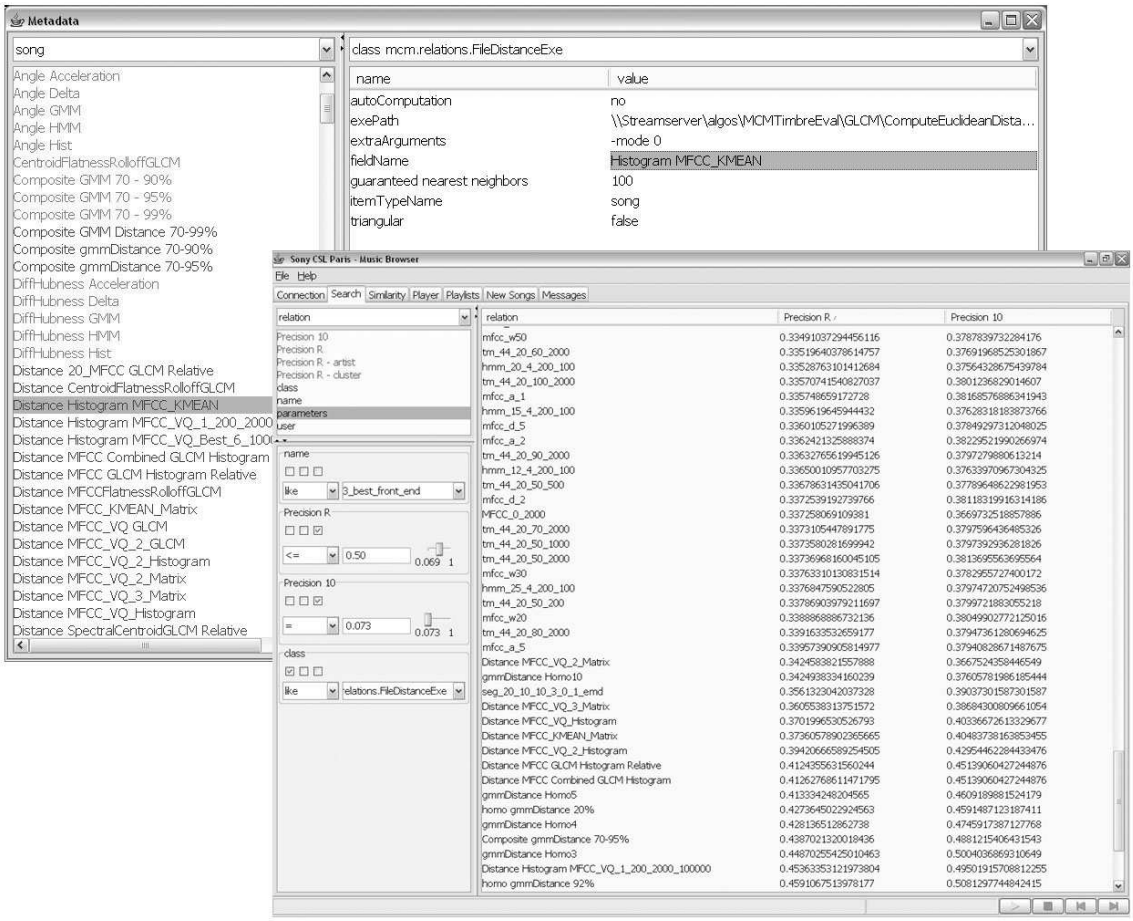

Figure 4.1: Screenshot of the Music Browser used as an experimentation platform in this study.

The “Search” panel of the Music Browser (in the foreground) shows the values of the metadata for the items of a given type. While traditional browsing scenarios would browse content items such as songs or artists, it is also possible to browse system items such as metadata. Here, we browse the collection of similarity measures. Furthermore, the genericity means that it is possible to create metadata *on* metadata. Figure 4.1 shows the example of precision values (`Precision R` and `Precision 10`) computed for each similarity measure. Such meta-metadata can naturally be edited in the “Metadata” panel described above and/or implemented anew as standard Java classes based on the MCM API.

4.3.2 Implementations

While many authors, such as (Pampalk, 2004), rely on Matlab implementations of the various algorithms, it appeared in this study that runtime performances were critical in order to enable the testing of many algorithm parameters over large ranges of values. Moreover, the above-mentioned need for a larger database architecture and metadata-management tools posed the additional problem of interoperability between the algorithm implementations and the Java-based tools such as MCM and the MB. Finally, an additional constraint is the flexibility to modify the implementations in order to test variants, which tends to favour proprietary implementations compared to third-party toolboxes.

Therefore, we investigated a number of alternative and faster implementations, both for feature extraction, distribution modelling and distance computations. Full details and comparison scores between custom and third party code, as well as different platforms (Matlab, C, Java) can be found in Appendix C. The best performing implementations, used in the following, are based on custom-code in C, called from Java by Java Native Interface (JNI).

4.3.3 Algorithms

Even with fast optimized implementations for distance computations, the task of finding the nearest neighbors (NN) of a given song among large sets of songs (typically several ten of thousands) is a very costly operation. This operation is needed both for the exploitation of a given similarity metric ("find me songs that sound like X") and for the repetitive evaluation of algorithmic variants that we propose to do in this study. This performance bottleneck is one of the principal reasons for the lack of systematic evaluation found in the literature.

One approach for speeding-up NN search is to use pre-built index structures, such as B+-trees or KD-trees (Samet, 1989) in euclidean spaces, or M-tree (Ciacca et al., 1997) and the Multi-vantage point (MVP) tree (Bozkaya and Ozsoyoglu, 1999) for metric spaces. Most of these indexes rely on the verification of the triangular inequality by the distance measure.

For the purpose of this study, we developed a generic algorithm for fast NN search in metric spaces which relies neither on an index structure, nor on the verification of the triangle inequality

by the distance measure. The algorithm exploits an intrinsic property of the class of similarity algorithms that we study here: all exhibit a *precision-cputime tradeoff* for some parameter p (*tradeoff parameter*), i.e. for which both the precision and the cputime increase with p. It uses successive approximations of the measure to compute more and more expensive measures on smaller and smaller sets. Full details about the algorithm can be found in Appendix D. We achieve speed improvement factors as high as 30, while still preserving more than 98% precision. These results are instrumental in the feasability of the large-scale evaluation reported here.

4.4 Results

4.4.1 Best Results

A complete account of the results of the evaluation of the variants listed in Section 4.1 can be found in Appendix B. We give here the main conclusions of the experiment:

- by hill-climbing the space of timbre models, we are able to increase the precision by more than 15% (absolute) over the original measure we introduced in Aucouturier and Pachet (2002b), to a maximum of 65,2% *R*-precision.
- the best performing measure is a fine-tuned version of the prototypical algorithm described in Chapter 3.1. It compares GMMs of MFCCs with Monte-Carlo approximation of the Kullback-Leibler distance. The optimal number of MFCCs and GMM components are 20 and 50 respectively, and MFCCs are appended with their 0^{th} order coefficient.
- among common speech processing front-ends, delta coefficients and 0^{th} order MFCC increase the precision by an unsubstantial extra 2% (absolute).
- dynamic modeling with hidden Markov models, and more complex processing inspired e.g. by image texture analysis do not increase the precision any further.

The optimal measure as found by this experiment and reported in Aucouturier and Pachet (2004a) was implemented by Elias Pampalk to win the “Artist Classification” contest at the ISMIR

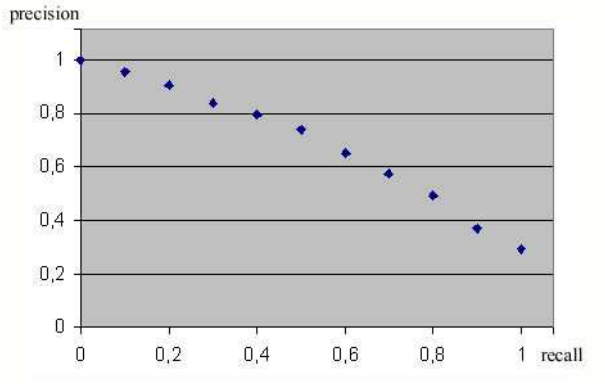

Figure 4.2: Precision vs Recall graph for the best measure

2004 Audio Description Contest (see `http://ismir2004.ismir.net/ISMIR_Contest.html`).

4.4.2 Significance

Figure 4.2 shows the precision-recall curve of the best measure (using 20MFCCs + 0th order coefficient + 50 GMMs). It appears that the precision decreases linearly with the recall rate (with a slope of about -5% per 0.1% increase of recall). This suggests that the *R*-precision value is a meaningful metric to compare variants to one another, and gives a good indication of how the algorithms behave in more realistic applications contexts (e.g. "give me the 5 nearest neighbors of this song").

While this 63% of precision may appear poor, it is important to note that our evaluation criteria necessarily underestimates the quality of the measure, as it doesn't consider relevant matches that occur over different clusters (*false negatives*), e.g. a Beethoven piano sonata is timbrally close to a jazz piano solo).

We should also emphasize that such an evaluation qualifies more as a measure of relative performance ("is this variant useful ?") rather than as an absolute measure. It is a well-known fact that precision measures depend critically on the test corpus and on the actual implementation of the evaluation process.

We do not claim that these results generalize to any other class of music similarity/classification/identification problems. However, as mentioned in Chapter 2, very many high-

level music description algorithms, such as genre classification, rely on the same paradigm for polyphonic timbre modelling than we examine here. While the evaluation and comparison of such systems require to also integrate higher-level correlations and groundtruth such as the categorical semantics ("how close is rock from pop"), we believe that our results, based on an explicit modelling of timbre similarity, contribute to a better understanding of the low-level aspects of some of these higher-level algorithms. Moreover, as will be seen in the next sections, the explicit examination of timbre similarity reveals a number of qualitative problems which are likely to also affect higher-level music description algorithms, for which they would be more difficult to diagnose.

4.4.3 Dynamics don't improve

A very notable result of this experiment is that classical pattern recognition extensions that take the data dynamics into account surprisingly fail to improve the precision of the models. This is notably true for computing first order derivatives of the features (so called "delta coefficients"), computing mean and variance of the features on intermediate-size "texture windows", using dynamical models such as recurrent neural networks or hidden Markov models, and second-order statistical analysis inspired by image texture classification. Full details about the evaluation of each of these algorithms can be found in Appendix B. Table 4.1 gives a subset of the evaluation scores achieved by such variants on the test database. This is a surprising observation, which is at odds with experimental evidence on the perception of monophonic instrument notes. The next chapter reports on an experiment aiming at better understanding this result.

4.4.4 "Everything performs the same"

The experiment show that, except a few critical parameters (sample rate, number of MFCCs), the actual choice of parameters and algorithms used to implement the similarity measure make little difference if any. We notice no substantial improvement by examining the very many variants investigated here : Complex dynamic modelling performs the same as static modeling. Complex front-ends, like spectral contrast, performs the same as basic MFCCs. Complex distance measures,

Table 4.1: *R*-precision scores achieved for the best algorithm as well as a number of extensions aiming at better modelling the dynamics of the data.

algorithm	*R*-Precision
best (20 MFCC + 50-state GMM)	0.65
Delta $\Theta = 10$	0.60
Acceleration $\Theta = 10$	0.610
Delta $\Theta = 2$	0.62
Delta $\Theta = 5$	0.62
Acceleration $\Theta = 5$	0.62
Acceleration $\Theta = 1$	0.63
Delta $\Theta = 1$	0.63
Texture Window $w_t = 10$	0.64
Texture Window $w_t = 20$	0.65
HMM (5 states)	0.62
HMM (10 states)	0.63
HMM (20 states)	0.62
HMM (30 states)	0.44
Texture CM (GLA-quantized)	0.48
Texture CM (LVQ-quantized)	0.44

such as EMD or ALA as reported in Berenzweig et al. (2003), performs the same as Monte Carlo, or even simpler centroid distances as also reported in Berenzweig et al. (2003). This behaviour has been mentioned before in the published partial comparisons between existing distance measures: Baumann (Baumann and Pohle (2003)) compares Logan and Salomon (2001), Aucouturier and Pachet (2002b) and Baumann (2003) and observes that “the different approaches reach similar performance”. Pampalk in Pampalk et al. (2003) remarks that the cluster organization of Logan and Salomon (2001) and Aucouturier and Pachet (2002b) are similar. Berenzweig et al. in Berenzweig et al. (2003) also conclude that the “different training techniques for GMMs (Kmeans or EM)” and “MFCC or anchor space feature achieve comparable results”.

4.4.5 Existence of a glass ceiling

The experiments reported here also suggest that the precision achievable by variations on the same classical pattern recognition scheme adopted by most contributions so far (including ours)

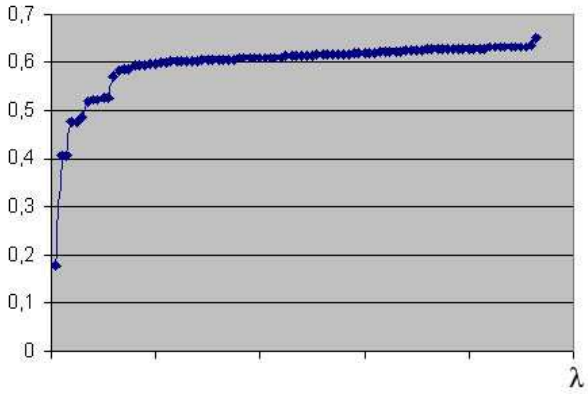

Figure 4.3: Increase in *R*-precision over the whole parameter space used in this paper

be bounded. Figure 4.3 shows the increase in *R*-precision achieved by the experiments in this chapter, over a theoretical parameter space λ (which abstracts together all the parameters and algorithm variants investigated here). The curve shows an asymptotic behaviour at around 65% (although this actual value depends on our specific implementation, ground truth and test database).

This is at odds with the common assumption found in the literature that the reported error-rates are incidental, and that near-perfect results would just extrapolate by fine-tuning the algorithm's parameters.

Obviously, this chapter does not cover all possible variants of the same pattern recognition scheme, as described in Chapter 4. Notably, one should also evaluate other low-level frame representations than MFCCs, such as LPCs or wavelets, and feature selection algorithms such as discriminant analysis. Similarly, newer methods of pattern recognition such as support vector machines have proved interesting for music classification tasks (Li and Tzanetakis (2003); Maddage et al. (2003)) and could be adapted and tested for similarity tasks. However, the set of features and variants used here, as well as the investigation of different modelling strategies (static vs dynamic, parametric vs non-parametric) is likely to capture most of the aspects covered by other variants. This suggests that the "glass ceiling" revealed in Figure 4.3 may also apply for further implementations of the same kind.

4.4.6 False Positives are *very* bad matches

Even if the *R*-precision reported here does not account for a number of *false negatives* (songs of different clusters that actually sound the same), the manual examination of the best similarity measure shows that there also remain some *false positives*. Even worse, these bad matches are not "questionably less similar songs", but usually are *very* bad matches, which objectively have nothing to do with the seed song.

We show here a typical result of a 10-nearest neighbors query on the song *HENDRIX, Jimi - I Don't Live Today* using the best set of parameters found above :

1. HENDRIX, Jimi - I Don't Live Today
2. HENDRIX, Jimi - Manic Depression
3. MOORE, Gary - Cold Day in Hell
4. HENDRIX, Jimi - Love or Confusion
5. MITCHELL, Joni - Dom Juan's Reckless Daughter
6. CLASH, The - Give Them Enough Rope
7. CLASH, The - Stay Free
8. MARDI GRAS BB - Bye Bye Babylon
9. HENDRIX, Jimi - Hey Joe
10. HENDRIX, Jimi - Are You Experienced

All songs by *Hendrix*, *Moore* and *The Clash* sound very similar, consisting in the same style of rock electric guitar, with a strong drum and bass part, and strong, male vocals. However, the song by *Joni Mitchell* ranked in 5th position is a calm folk song with an acoustic guitar and a female

singer, while the 8th item is a big band jazzy tune. Similar bad matches are sometimes reported in the literature, e.g. in Pampalk et al. (2003) “a 10-second sequence of *Bolero* by *Ravel* (Classical) is mapped together with *London Calling* by *The Clash* (Punk Rock)”, but most of the times, the very poor quality of these matches is hidden out by the averaging of the reported results.

4.4.7 Existence of hubs

Interestingly, in our test database, a small number of songs seems to occur frequently as false positives. For instance, the fifth neighbor of the Hendric query given above, `MITCHELL, Joni - Don Juan's Reckless Daughter`, is very close to 1 song out of 6 in the database (57 out of 350). However, the cluster corresponding to it artist only contains 9 songs, i.e. this specific song occurs more than 6 times more than it should. Among all its occurrences, many are likely to be false positives.

This suggests that the 35% remaining errors are not uniformly distributed over the whole database, but are rather due to a very small number of songs, which we may call “hubs”, which are close to all other songs. These hubs are especially intriguing as they usually stand out of their clusters, i.e. other songs of the same cluster as a hub are not usually hubs themselves.

Chapter 6 is devoted to a further study of the phenomenon of hubs. We conduct a number of experiments which shows that this is not a boundary effect of our small-database evaluation, but rather probably a general structural property of the class of algorithms investigated in this work.

Chapter 5

Experiment 2: The Usefulness of Dynamics

This chapter further examines one of the most surprising results of **Experiment 1**, namely that models that account for the time dynamics of the features are at best equivalent to simpler static models. This contradicts experimental data on the perception of individual instrument notes. We propose three possible causes for the difficulty of modelling dynamics of full songs, and discriminate between them by comparing the performance of dynamical algorithms on several specially designed datasets. We conclude that the main cause of the difficulty of modelling dynamics is the polyphonic nature of the data.

5.1 The paradox of Dynamics

One of the surprising conclusions of the evaluation made in Chapter 4 is that classical pattern recognition extensions that take the data dynamics into account surprisingly fail to improve the precision of the models. This is notably the case for computing first order derivatives of the features (so called "delta coefficients") as introduced by Furui (1986), computing mean and variance of the features on intermediate-size "texture windows" (Tzanetakis and Cook, 2002) or using dynamical models such

as recurrent neural networks or hidden Markov models (Rabiner, 1989).

This is surprising, as we have already noted that static models consider frame-permutations of the same audio signal as identical, while this has a critical influence on their perception. Moreover, psychophysical experiments as we reported in Chapter 2 have established the importance of dynamics, notably the attack time and fluctuations of the spectral envelope, in the perception of individual instrument notes.

This probably means that existing pattern-recognition methods do not address the right structural time-scale, or are unable to learn it from data that is either too sparse or too variable. It has been recently argued in the MIR community (Ellis, 2005) that although hidden Markov modelling typically achieves a better fit to the modelled data (in terms of likelihood) than static models such as GMMs, it does so by capturing non-discriminative baseline temporal correlation, which is not so different among different songs or genres. The poor improvement, if any, achieved by state-of-art dynamical models has recently been confirmed by Scaringella and Zoia (2005) on the related task of musical genre classification.

5.2 Hypothesis

There are 3 main hypothetical causes that explain the difficulty of modelling dynamics in the case of polyphonic timbre textures:

H1 Either the dynamics of timbre frames are impossible to capture at the time-scale of an individual note, e.g. because there is too much timing variation from note to note. This is improbable, as success in doing so has been reported notably in computer-based psychophysics analysis of timbre perception (Krimphoff et al., 1994) or instrument note recognition (Dubnov and Fine, 1999; Eronen, 2003).

H2 Either its is the dynamics of *polyphonic* timbre frames that is difficult to model. We have already noted the problems of spectral masking and asynchronicity of several concurrent sound sources, and how they defeat naïve analysis generalized from the monophonic case.

H3 Or it is the dynamics of *successive notes* at the time-scale of a phrase or a full song that are difficult to model. It is unclear e.g. whether a HMM attempts to capture fine-grained dynamics such as the succession of transient and steady-state inside individual notes or rather longer-term structure like the succession of different instrument timbres.

5.3 Method

We propose to discriminate between these 3 hypotheses by testing the performance of dynamical algorithms on two databases of individual sound samples:

- one composed of clean, monophonic individual instrument notes (DB1)
- the other obtained from a polyphonic, real-world recording (DB2)

The comparison of the performance of dynamical modelling against static modelling in both contexts has the potential to disprove some of the above hypotheses. Evidence that dynamical modelling performs constantly worse than static modelling on both databases would support H1 (and be at odds with previous findings from the literature). H2 will be supported should dynamical modelling perform better than static on the monophonic dataset (DB1), but not on the polyphonic one (DB2): this would mean that polyphony ruins attempts at modelling dynamics even within the constrained time-scale of individual notes. Finally, evidence that dynamical models overperforms static models for both databases, but not for textures made of successive notes as we established in Chapter 5, would indicate that the critical factor is the existence of the longer-term structure of e.g. phrase rhythm and instrument changes.

5.3.1 Databases

Table 5.1 and 5.2 describe the contents of both databases. DB1 was obtained as an extract of the IRCAM “Studio On Line” database, made available in the context of the Cuidado European Project (Pachet et al., 2004). It contains 710 short sound samples, categorized in 16 classes corresponding to

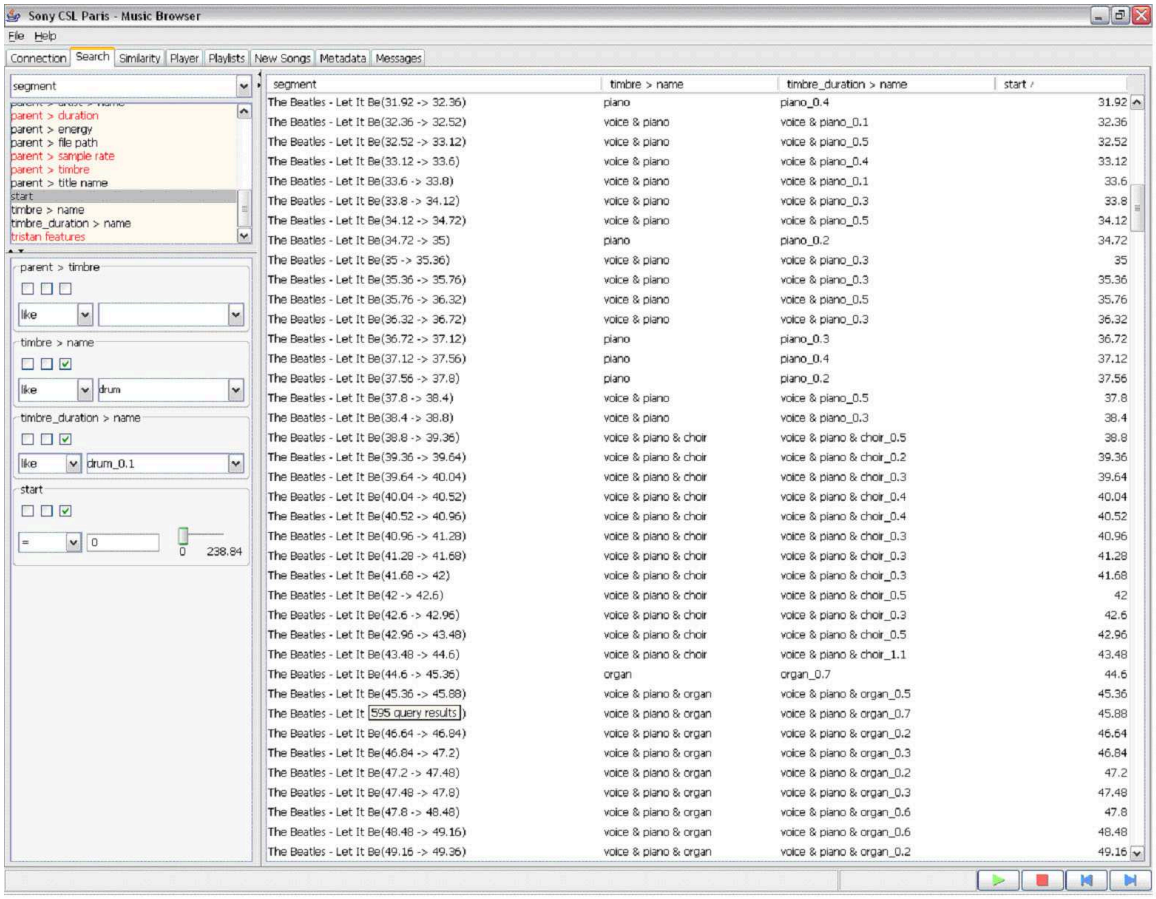

Figure 5.1: Groundtruth for polyphonic samples

the instrument used for their recording. DB2 consists of sound samples obtained from the automatic analysis of the song *The Beatles - Let it be*, using the automatic segmentation algorithm described in Appendix E. The process yielded 595 samples, which were manually clustered and categorized into 16 categories, corresponding to the different mixtures of sound sources occurring in the song (a few samples were discarded because they were either too short, or difficult to categorize). Figure 5.1 shows a screenshot of the Music Browser application, which was used to annotate DB2.

We remark that both database have the same number of classes, and roughly the same size, which makes their comparison quite reliable. However, DB2 being obtained with automatic segmentation, samples from the same category may have quite different durations. This may be detrimental to dynamical algorithms, which may match samples of the same duration across different

categories. To control this effect, we create a third database, DB3, which contains the same samples as DB2, but sub-categorizes the timbre categories according to the samples' duration: samples that were categorized as e.g. *Piano* in DB2 are categorized in DB3 as one of {*Piano_100*, *Piano_200*, ...}, where {100,200,...} denotes the duration (in ms) of the sample (averaged to the nearest multiple of 100ms). The typical sample duration being between 50ms and 1 sec., this creates up to 10 time-indexed sub-class per original instrument class in DB2. Figure 5.1 shows both the DB2 and DB3 labels for an extract of the database.

Table 5.1: Composition of the monophonic database.

Class	Number of instances
Accordion	37
Alto violin	51
Bass	50
Bassoon	39
Cello	48
Clarinet	44
Flute	38
Guitar	66
Harp	40
Horn	78
Oboe	36
Sax	32
Trombone	38
Trumpet	32
Tuba	35
Violin	46
Total	710

5.3.2 Algorithms

17 algorithmic variants were implemented for each database and their results compared. Table 5.3 describes the parameters of each variant.

The use of dynamic information is embodied by 3 algorithmic variants based on Dynamic Programming (DP, see e.g. Crochemore and Rytter (1994)). DP is typically used for aligning or com-

Table 5.2: Composition of the polyphonic database.

Class	Number of instances
Drums	119
Electric Guitar	148
Electric Piano	26
Organ	55
Organ & Drums	32
Piano	191
Piano & Tchak	53
Tutti 1	142
Voice & Bass & Drums	71
Voice & Organ & Drums	22
Voice & Piano	116
Voice & Piano & Choir	12
Voice & Piano & Organ	10
Voice & Piano & Tchak	39
Voice & Tutti 1	107
Voice & Tutti 1 & Electric Guitar	43
Total	595

puting the distance between 2 sequence of symbols, such as text, protein or DNA sequences. It was also used e.g. in Smith et al. (1998) for comparing sequences of musical notes. DP relies on a symbol-distance, which measures the distance between duplets of symbols in the alphabet (which may have infinite size), and an edit-operation cost, which penalizes alignment changes in the sequence (e.g., deletion, insertion, substitution). In our case, we compare sequences of MFCC frames, using the euclidean distance as symbol distance, and compare 3 values for the edit cost $\{10, 100, 1000\}$. The smaller the edit-cost, the more tolerant the measure is to modifications of the time arrangement of successive MFCC frames. This makes it possible to align close MFCC frames at different positions within the samples, i.e. to match sound samples of the same timbre, but with very different duration. However, this also increases the number of false positives. DP can be viewed as a manual equivalent of decoding the sequence with an a priori trained HMM. Note however that HMM-based similarity (as used for full songs in Chapter 4 and Appendix B.5.3) was impossible to use in the context of short samples, because of the lack of training data: a typical sample has a duration of 200 ms, which amounts to 10 frames.

Table 5.3: Description of the algorithms used to compare monophonic and polyphonic sample similarity

DynProg MFCC (edit 10)	Dynamic Programming Comparison of MFCC frames (using an edit cost of 10)
DynProg MFCC (edit 100)	Same as above with edit cost of 100
DynProg MFCC (edit 1000)	Same as above with edit cost of 1000
MFCC_MP7_RMS 1G	Monte-Carlo KL comparison of Gaussian Mixture model (using 1 gaussian component) of feature vectors composed of MFCCs (dim 20), MP7 Spectral descriptors (dim 7) and RMS value (dim 1)
MFCC_MP7_RMS 2G	Same as above with 2 gaussian components
MFCC_MP7_RMS 3G	Same as above with 3 gaussian components
MFCC_MP7_RMS 4G	Same as above with 4 gaussian components
MP7 1G	Monte-Carlo KL comparison of Gaussian Mixture model (using 1 gaussian component) of feature vectors composed of MP7 Spectral descriptors (dim 7)
MP7 2G	Same as above with 2 gaussian components
MFCC_MP7 1G	Monte-Carlo KL comparison of Gaussian Mixture model (using 1 gaussian component) of feature vectors composed of MFCCs (dim 20) and MP7 Spectral descriptors (dim 7)
MFCC_MP7 2G	Same as above with 2 gaussian components
MFCC_RMS 1G	Monte-Carlo KL comparison of Gaussian Mixture model (using 1 gaussian component) of feature vectors composed of MFCCs (dim 20) and RMS value (dim 1)
MFCC_RMS 2G	Same as above with 2 gaussian components
MFCC 1G	Monte-Carlo KL comparison of Gaussian Mixture model (using 1 gaussian component) of feature vectors composed of MFCCs (dim 20)
MFCC 2G	Same as above with 2 gaussian components
Mean MFCC Euclidean	Euclidean comparison of the mean of feature vectors composed of MFCCs (dim 20)
Jehan Euclidean	Euclidean comparison of the concatenation of the mean of feature vectors composed of MFCCs (dim 20), and a set of global temporal shape descriptors (dim5)

We compare these dynamical algorithms to a number of static algorithms, based on combinations of features such as MFCC, energy (root-mean square) or Spectral MP7 descriptors (i.e. `SpectralCentroid`, `SpectralSpread`, `SpectralKurtosis`, `SpectralSkewness`, `SpectralFlatness`, `SpectralRolloff`, `SpectralFlux`). Features are compared using simple average comparison with euclidean distance, or Gaussian Mixture models (i.e. average *and* variance) with up to 4 gaussian components. Note that similarly to HMM-based processing, greater numbers of components (such as 50 as used for full songs) could not be tested because of the lack of training data in individual samples.

Finally, a hybrid algorithm inspired by Jehan (2004), compares a feature vector composed of the average of the MFCCs and a set of global descriptors describing the temporal shape of the samples: normalized loudness at onset and at offset, maximum loudness and relative location of the maximum loudness.

5.3.3 Evaluation Procedure

The algorithms are compared by computing their precision after 10 documents are retrieved, and their R-precision, i.e. their precision after all relevant document are retrieved. Each value measures the ratio of the number of relevant documents to the number of retrieved documents. The set of relevant documents for a given sound sample is the set of all samples of the same category than the seed. This is identical to the methodology used for Experiment 1 (Chapter 4).

5.4 Results

Table 5.4 shows the evaluation scores of the algorithms described above on both databases DB1 and DB2. One can see that dynamic algorithms perform up to *10% better* (absolute) than static algorithms on the monophonic database. This establishes that the dynamic evolution of instantaneous features are an important factor for timbre similarity. This confirms the findings of both psychophysical experiments on the perception of instrument timbre, and a number of automatic

instrument classification systems. The best static performances on DB1 are obtained with fairly involved variants, which typically rely on concatenation of several features, and large Gaussian Mixture Models.

However, dynamic algorithms perform nearly *10% worse* than their static equivalent on the polyphonic database DB2. It also appears that the best polyphonic performance is achieved with the most simple static algorithms, such as euclidean comparison of the simple average of MFCCs. Notably, while increasing the number of gaussian components for the `MFCC_MP7_RMS` family of algorithms constantly increases the precision on the monophonic dataset (from 0.62% to 0.64 % R-precision), the same operation degrades the precision (from 0.51% to 0.42%) in the polyphonic case.

Results on the duration-indexed version of the polyphonic database (DB3) confirm the fact that dynamical algorithms are helped by keeping the duration constant within a class. Conversely, static algorithms that do not consider duration are penalized by blindly returning samples which may be of the correct DB2 class, but not in the correct DB3 class. However, the performance of dynamical algorithms on DB3 remains more than 25% worse than the static performance on DB2, which shows that, even at constant duration, dynamical algorithms are poor at capturing essential feature dynamics.

Overall, the observation that dynamic algorithms overperform their static counterparts on DB1, but are ranked in inverse order on DB2 gives strong evidence that polyphony ruins attempts at modelling dynamics even within the constrained time-scale of individual notes. This conclusion therefore generalizes all the more so to sequences of notes, and explains the poor performance of dynamical algorithms for the timbre similarity of full songs.

Moreover, polyphony seem to make difficult the training of involved static algorithms such as several-component GMMs. These perform less accurately than simplistic euclidean comparison of the mean frame of each segment. As polyphonic samples tend to be longer than monophonic samples, this is not simply an effect of overfitting complex models to too little training data (see Section 3.2.2), but a property of the data itself. Polyphony, and notably the quasi-random superposition of

Table 5.4: Comparison of similarity methods for monophonic and polyphonic samples. Best scores for each dataset appear in bold.

Method	Monophonic (DB1)		Polyphonic (DB2)		Polyphonic2 (DB3)	
	P_{10}	P_R	P_{10}	P_R	P_{10}	P_R
DynProg MFCC (edit 10)	**0.76**	**0.46**	0.44	0.34	**0.24**	**0.22**
DynProg MFCC (edit 100)	**0.73**	**0.46**	0.37	0.27	**0.35**	**0.31**
DynProg MFCC (edit 1000)	**0.70**	**0.44**	0.31	0.17	**0.33**	**0.28**
MFCC_MP7_RMS 4G	0.64	0.34	0.42	0.31	0.12	0.12
MFCC_MP7_RMS 3G	0.63	0.34	0.45	0.32	0.12	0.12
MFCC_MP7_RMS 2G	0.62	0.33	0.47	0.35	0.14	0.13
MFCC_MP7_RMS 1G	0.62	0.35	**0.51**	**0.37**	0.15	0.14
MP7 1G	0.61	0.38	0.36	0.29	0.11	0.11
MFCC_MP7 1G	0.61	0.33	0.47	0.35	0.14	0.13
MP7 2G	0.61	0.38	0.36	0.29	0.11	0.11
MFCC_MP7 2G	0.59	0.31	0.43	0.33	0.12	0.12
Mean MFCC Euclidean	0.58	0.33	**0.50**	**0.39**	0.14	0.13
Jehan Euclidean	0.56	0.32	**0.49**	**0.38**	0.21	0.19
MFCC_RMS 2G	0.56	0.28	0.48	0.35	0.14	0.13
MFCC_RMS 1G	0.55	0.27	**0.50**	**0.37**	0.15	0.14
MFCC 2G	0.51	0.26	0.46	0.32	0.14	0.13
MFCC 1G	0.50	0.26	0.47	0.33	0.15	0.13

asynchronous sources in a given sound sample, probably creates a higher degree of variance from one sound sample to another than in the monophonic case. This effect could probably be limited if more data were available, e.g. in the context of classification where models are trained on a set of several songs, instead of individual songs as we do here for similarity.

Chapter 6

Experiments 3-8: Understanding Hubs

Probably the most important and novel finding of **Experiment 1** is that the class of algorithms studied in this work tend to create false positives which are mostly always the same songs regardless of the query. In other words, there exist songs, which we call *hubs*, which are irrelevantly close to all other songs. This phenomenon is reminiscent of other isolated reports in different domains, such as Speaker Recognition or Fingerprint Identification, which intriguingly also typically rely on the same pattern-recognition algorithms. This suggests that this could an important phenomenon which generalizes over the specific problem of timbre similarity, and indicates a general structural property of the class of algorithms examined here. This chapter reports on a number of experiments aiming at better understanding the nature and causes of hub songs.

6.1 Definition

In this chapter, we call *hub* a song which occurs frequently as a false positive according to a given similarity measure. This both implies that

1. a hub appears in the nearest neighbors of most songs in the database
2. most of these appearance do not correspond to any meaningful perceptual similarity.

Each condition in itself is not sufficient to characterize a hub:

1. A given song may occur very many times in the nearest neighbors of other songs, but this may not be a bug. Depending on the composition of a given databases, some songs may well approximate the perceptual center-of-mass of the database. For instance, it may be found that *A Hard Day's Night* by *The Beatles* is a song that bears close timbre similarity to most of 60's pop music, and therefore could be found to occur very frequently as a nearest neighbor to many songs in a database composed by a majority of Rock and Pop songs. However, in a classical music database, the same song would not be a hub.

2. A given song may be a false positive for a given seed song, i.e. be in the first nearest neighbors of the seed without any actual perceptual similarity. However, different songs may have different false positives. For instance, a given *Beethoven* piano sonata may be mismatched to an acoustic guitar piece, but not necessarily mismatched to other songs. A hub is a piece than is irrelevantly close to very many songs, i.e. a bug which is not local to only a few queries.

6.2 Why this may be an important problem

The existence of objects that tend to be very frequent false positives for pattern recognition algorithm has long been acknowledged in other domains that music. Biometric verification systems, such as fingerprints, but also speech and speaker recognition systems typically exhibit striking performance inhomogeneities among users within a population. The statistical significance of such critical classes of users, in the context of Speaker Verification, was formally shown in Doddington et al. (1998), by analysing population statistics based on the test data used for the NIST 1998 speaker recognition evaluation. This evaluation includes data from more than 500 speakers and recognition results from 12 systems. The paper established a speaker taxonomy in terms of animal names:

Goat Goat users are users that are very difficult to recognize, i.e. that are associated with a high rate of false reject.

Lamb Lambs are those users that are particularly easy to imitate. Randomly chosen users are exceptionally likely to impersonate a lamb.

Wolf Wolves are those users who are particularly successful at imitating other speakers. Their speech is exceptionally likely to be accepted as that of another speaker. (Figure 6.1)

Figure 6.1: So-called "wolf" speakers are exceptionally successful at imitating other speakers with speaker recognition systems. Drawing courtesy of Johan Koolwaaij, retrieved from http://www.ispeak.nl/

A complete analogy with this taxonomy would call a wolf a song which is constantly closer to a random song S than S is to itself. However, the Speaker Recognition *menagerie* is essentially pointing out the same phenomenon as the hubs observed with our timbre similarity measure: that high false positive rates are not uniformly distributed in the database, but manifests only in a small critical population.

Goats and Wolves users are critical to take into account to carefully evaluate system performance (Koolwaaij and Boves, 1997; Bimbot and Chollet, 1997). Workarounds have been designed to pragmatically improve system performance, such as the use of cross-validation with cohort speakers (Rosenberg et al., 1992) or separate processing of pre-filtered "lamb" speakers (Jin and Waibel, 2000). Nevertheless, the reason for the appearance of such classes has rarely been questioned, and

is generally thought to be an intrinsic property of human users*.

However, a recent study (Hicklin et al., 2005) in the context of fingerprint recognition suggests that these properties of wolfiness, goatness, etc.. are not intrinsic properties of the users themselves, but rather properties of the algorithms. The observation that we make here of the existence of "wolf songs', in the different context of music pieces, seems to corroborate this hypothesis. This is especially interesting as the techniques used for timbre similarity (namely variations on the GMMs of MFCCs) are typically similar to the ones employed in Speaker/fingerprint recognition systems. We will show in the remaining of this chapter that we can also observe that the hubness of a given song is algorithmic-dependent, and that the existence of such critical classes may indicate a more general structural property of pattern recognition techniques.

6.3 Measures of hubness

Several measures can be used to identify and quantify the "hubness" of a given song. We describe here two of such measures. Greater details and alternative measures are given in Appendix F.

6.3.1 Number of occurrences

A natural measure of the hubness of a given song is the number of times the song occurs in the first *n* nearest neighbors of all the other songs in the database. Table 6.1 shows a few songs in the test database along with the number of times they occur in the first 10 nearest neighbors over all queries (N_{10}). This illustrates the predominance of a few songs that occur very frequently. For instance, the first song, `MITCHELL, Joni - Don Juan's Reckless Daughter` is very close to 1 song out of 6 in the database (57 out of 350).

This measure has the following properties:

- Independent of distance: Being based on rank, the number of occurrences of a song is independent of the range of the values produced by a given distance measure. Therefore, it

*Note that this would be intriguingly true of human speech but also fingerprints, etc.

Table 6.1: 15 Most Frequent False Positives

Song	N_{10}	$card(C_S)$	$\frac{N_{10}}{card(C_S)}$
MITCHELL, Joni - Don Juan's Reckless Daughter	57	9	6.33
RASTA BIGOUD - Tchatche est bonne	30	7	4.23
MOORE, Gary - Separate Ways	35	9	3.88
PUBLIC ENEMY - Cold Lampin With Flavor	27	8	3.37
GILBERTO, Joao - Tin tin por tin tin	25	8	3.12
CABREL, Francis - La cabane du pêcheur	22	7	3.14
MOORE, Gary - Cold Day In Hell	27	9	3.0
CABREL, Francis - Je t'aimais	20	7	2.86
MOORE, Gary - The Blues Is Alright	25	9	2.77
MARDI GRAS BIG BAND - Funkin'Up Your Mardi Gras	19	7	2.71
RASTA BIGOUD - Kana Diskan	18	7	2.57
BRIDGEWATER, DD - What Is This Thing Called Love	30	12	2.5
Frehel - A la derive	20	8	2.5
ADAMS, Bryan - She's Only Happy When She's Dancin'	20	8	2.5
MITCHELL, Joni - Talk To Me	22	9	2.44

can be used to compare hubs appearing with different algorithms, which we will do e.g. in Section 6.6.

- Dependant on database: The total number of occurrence of a song is composed both of true and false positives. As explained earlier, only the latter are characteristic of a hub. This metric therefore is conservative in the sense that if a high number of occurrence is observed for a given song in an arbitrary database, it is difficult to conclude whether it is indeed a hub (i.e. that most of these occurrences correspond to false positives) or a perceptual center-of-mass (i.e. most of the occurrences are true positives).

- Constant-sum: An important property of the number of n-occurrences N_n of a song is that the sum of the values for all songs is constant given a database. Each query only gives the opportunity for n occurrences to the set of all the other songs, such that the total number of n-occurrences in a given $\mathcal{N}$-size database is $n * \mathcal{N}$. Therefore, the mean n-occurrence of a song is equal to n, independently of the database and the distance measure. Alternatively, if we assume that the distance engenders a uniform, random distribution of the song, a given

song has the probability $p = \frac{n}{N}$ to occur in the n-nearest neighbor of another song, which indeed gives an expected number of occurrences $E(N_n) = \mathcal{N} * p = n$. Table 6.2 illustrates the experimental verification of this property (constant mean) for several distance algorithms.

Table 6.2: Comparison of mean nb of occurrences and mean neighbor angle for songs in the test database, for several distance algorithms

Measure	GMM	HMM	Delta	Acceleration	Histogram
N_{100}	100.2	98.7	99.4	99.4	99.6
Neighbor angle (degrees)	58.8	55.6	58.3	57.9	59.9

- Descriptive statistics: This has the notable consequence that the mean value of N_n is useless to measure the influence of a given algorithm on the global hubness of a database. One has to look for other descriptive statistics, such as the variance of the distribution of occurrences, or the number of songs with more than a given number of occurrences.

6.3.2 Neighbor angle

An operational definition of a hub is that it is a song H which is found to be "close" (though not perceptually) to duplets of songs A and B which themselves are (perceptually) distant from one another. Note that songs close to many songs which are themselves close to one another would indicate an acceptable "center-of-mass" situation. Therefore, the hubness of song H can be estimated by comparing its distances to its neighbors $d(H, A)$ and $d(H, B)$ on the one hand, and the distance between the neighbors $d(A, B)$ on the other hand. Equivalently, on can measure the angle θ formed by the segments $[H, A]$ and $[H, B]$. As seen in Figure 6.2, the angle θ can be expressed in terms of $d(H, A)$, $d(H, B)$ and $d(A, B)$.

$$d = d(H, B) \sin\theta = d(A, B) \sin\alpha \tag{6.1}$$

$$d' = d(H, A) - d(H, B) \cos\theta = d(A, B) \cos\alpha \tag{6.2}$$

$$\Rightarrow d^2 + d'^2 = d(H, A)^2 + d(H, B)^2 - 2d(H, A)d(H, B) \cos\theta = d(A, B)^2 \tag{6.3}$$

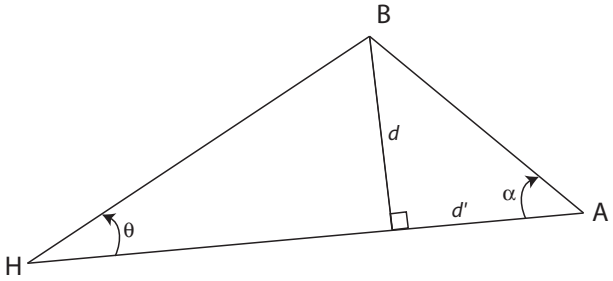

Figure 6.2: The neighbor angle θ can be expressed in terms of $d(H, A)$, $d(H, B)$ and $d(A, B)$.

and therefore

$$h_2(H, A, B) = \cos\theta = \frac{d(A, B)^2 - d(H, A)^2 - d(H, B)^2}{2d(H, A)d(H, B)} \tag{6.4}$$

This is computed for a given song H by drawing a large number of successive duplets of neighbors (A, B) (such that $A \neq B \neq H$), and computing the mean value of $h_2(H, A, B)$. We use 1000 successive random draws.

Measures of neighbor angle have the following properties:

- Independent of database: Unlike measures of the number of occurrence of a song, distance-based metrics such as neighbor angle are independent of the possible perceptual clusters of a given database. Thus they can be used to compare algorithms on different databases.

- Dependant on algorithm: Neighbor angle is dependent on the discrimination capacity of the distance, i.e. the typical distance ratio between what can be considered a close distance, and what can be considered a large distance.

- Constant-sum: An important property of the neighbor-angle value is that, like the number of n-occurrences N_n of a song, the sum of the values for all songs is constant given a database size. This directly derives from the fact that the angles of a triangle sum to π radians (in a euclidean geometry - which is only approximated here in the general case [†]). Given a set of $\mathcal{N}$ points, the number of angles whose vertex is a given point X, and are formed by the lines

[†]This is not a problem in terms of cognitive modelling. Tversky (1977) showed that measures of similarity that conform to human perception do not satisfy the usual properties of a metric, notably symmetry and triangular inequality

from X to the $\mathcal{N}-1$ other points, is equal to the number of combinations of 2 points within $\mathcal{N}-1$, i.e. $C^2_{\mathcal{N}-1}$. There are $\mathcal{N}$ possible vertex X for such angles, thus there are a total of $\mathcal{N}C^2_{\mathcal{N}-1} = \frac{n(n-1)(n-2)}{2}$ angles formed between the $\mathcal{N}$ points. It is easy to see that $n(n-1)(n-2)$ is divisible by $3 \forall n$. Hence, these angles can be clustered by triplets, so that their supporting lines form a triangle, and thus sum to π. Therefore, the sum of all angles formed between $\mathcal{N}$ points equals $\frac{\mathcal{N}C^2_{\mathcal{N}-1}}{3}\pi$. Table 6.2 illustrates the experimental verification of this property (constant mean) for several distance algorithms. The deviation of the mean angle from the theoretical value 60° is both explained by the statistical approximation of the computation of the angles and by the possible non-euclideanity of the underlying geometry.

- Descriptive statistics: This has the notable consequence that the mean value of the neighbor angle is useless to measure the influence of a given algorithm on the global hubness of a database. Like for occurrence values, one has to look for other descriptive statistics, such as the number of songs with a mean angle greater than a given limit.

6.3.3 Correlation between measures

As can be seen in Figure 6.3, there is a nearly logarithmic dependency between the number of occurrences of a given song and its mean neighbor angle. Table 6.3 shows the linear correlation scores between the logarithm of N_{100} and the neighbor angle measure, for several models. The best fits are achieved for the static models, both parametric and non-parametric. Dynamic models tend to create more outlier points in the scatter plots, which reduce the correlation scores. It appears that hub songs tend to be associated to higher values of neighbor angle. However, the logarithmic dependency makes it difficult to distinguish songs with number of occurrences in the range 100–200 using their value of neighbor angle. Therefore, in the remaining of the chapter, the rank-based measure will be preferred when comparing different settings in the same database.

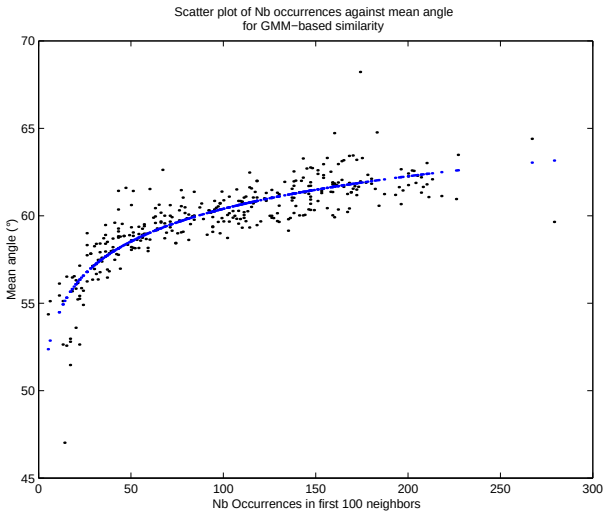

Figure 6.3: Scatter plot between number of 100-occurrences N_{100} and mean neighbor angle for a distance based on GMM.

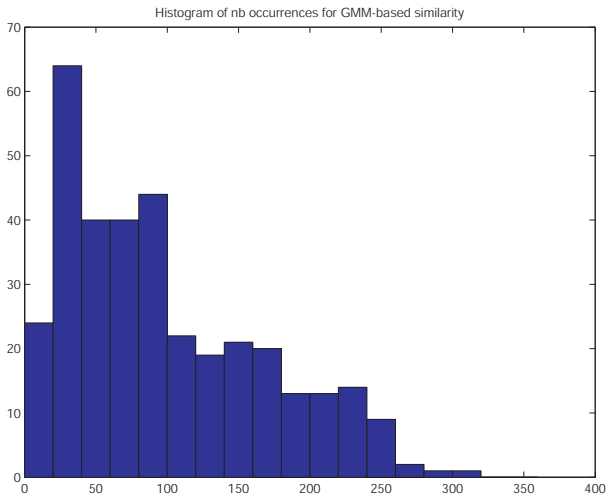

Figure 6.4: Distribution of the songs according to their number of 100-occurrences in the 360-song test database, with a GMM-based distance.

Table 6.3: Correlation between the logarithm of the number of 100-occurrences N_{100} on the one hand, and mean neighbor angle on the other hand, for various models

GMM	HMM	Delta	Acceleration	Histogram
0.85	0.76	0.73	0.74	0.93

6.4 Power-law Distribution

In this section, we examine the distribution of hub songs in a typical database. Figure 6.4 shows the distribution of the songs in the test database according to their number of 100-occurrences, for a GMM-based distance. We observe the following facts:

- The distribution decreases regularly from 50 to the maximum possible value 360 (not reached). It is skewed left-wise with respect to the theoretical mean value (100).

- The majority of songs have a N_{100} value in the range $[50 - 100]$, which correspond to an expectable standard behaviour: songs in a given cluster are typically close to songs from the same cluster (say around 15), and from songs from 2-3 neighboring clusters.

- About 15% of the songs exhibit a N_{100} value larger than 200 (out of a maximum possible 360). Such songs can reasonably be labelled as hubs.

- Hubness is not a discrete boolean property, but rather a continuous variable. The database exhibits a continuum of hubs of varying importance, with a couple of songs having a N_{100} value larger than 300, but also intermediate values in the range $[100 - 200]$.

In order to better estimate the hub distribution, we implement the best achieving measure (GMM-based) on a much larger database that was assembled for the Cuidado project (Pachet et al., 2004). The database currently contains 15,460 mp3 files. Figure 6.5 shows the distribution of songs in the Cuidado database according to their number of 100-occurrences. One can observe that a few songs get upward of 2000 occurrences, whereas most songs only have around a few hundred occurrences (more than 6,000 songs have between 150 and 160 occurrences). The distribution is strongly

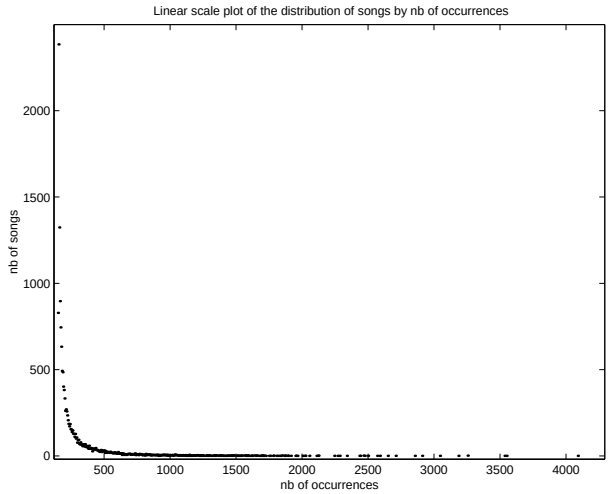

Figure 6.5: Distribution of the songs according to their number of 100-occurrences in the Cuidado 15,000-song database, with a GMM-based distance.

reminiscent of a power-law:

$$P[X = x] = x^{-\gamma} \tag{6.5}$$

Figure 6.6 shows the same plot than Figure 6.5, but on a log-log scale the same distribution shows itself to be linear. This is the characteristic signature of a power-law. To get a proper fit, the N_{100} were binned into exponentially wider bins (thus appearing evenly spaced in the log domain). A nearly linear relationship extends over 4 decades ($[1-10^4]$) songs, which is why such distribution have been called “scale-free”, or lacking a “characteristic length scale”. This means that no matter what range of x one looks at, the proportion of small to large events is the same, i.e. the slope of the curve on any section of the log-log plot is the same.

A power-law describes a situation where small occurrences are extremely common, whereas large instances are extremely rare. Many man-made and naturally occurring phenomena, including city sizes, incomes, word frequencies, and earthquake magnitudes, are distributed according to a power-law distribution (Bak, 1996). Recently, attention has turned to the internet which seems to display quite a number of power-law distributions: the number of visits to a site, the number of pages within a site (Huberman and Adamic, 1999), and the number of links to a page (Albert et al.,

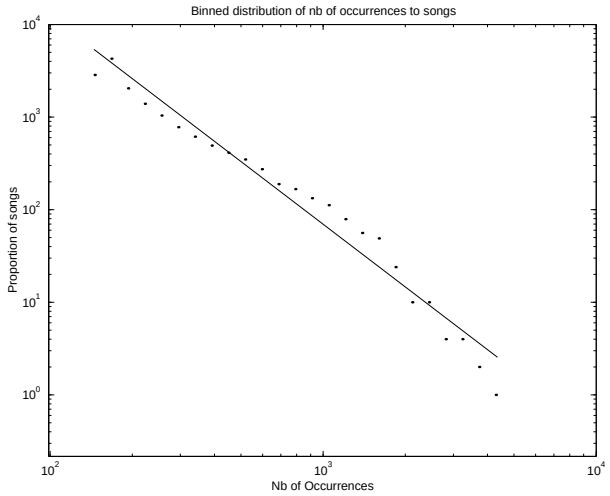

Figure 6.6: Log-log plot of the distribution of the songs according to their number of 100-occurrences in the Cuidado 15,000-song database, with a GMM-based distance. The distribution is approximately linear, which indicates a power-law.

1999), to name a few. Similarly, scale-free distributions have been observed in musical data, notably in networks of artists that co-occur in playlists from specialized websites (Cano and Koppenberger, 2004).

For all these reasons, the scale-free distribution of networks of timbrally similar songs is a remarkable, but not utterly surprising phenomenon. If all timbre distances were perceptually relevant ("no bugs"), then it would an acceptable conclusion that some songs be more "prototypical" than others, thus translating the distribution of musical and social influences and communities inherent to possibly every human activity. However, as already noted, what we observe here is a distribution of algorithmic bugs rather than a self-organization of a music space: The most connected songs (extreme hub songs that are close matches to more than a third of a given database) typically appear as the nearest neighbors of songs to which they do not bear any perceptual similarity. It is yet unclear whether the scale-free distribution that we observe here is

- the result of a scale-free organisation of an ideal perceptual distance measure, which is being polluted by measurement errors
- the result of a non remarkable ideal distribution, polluted by a scale-free distribution of false-

positives

- or both

The influence of measurement errors on scale-free distributions could be studied e.g. in the light of recent results on the robustness of experimental topological analysis of protein interaction networks (Lin and Zhao, 2005).

6.5 Experiment 3: Features or Model ?

6.5.1 Hypothesis

In this section, we investigate whether hubs are a consequence of poor featural representation of the frames of audio data. We test the hypothesis that hubs exist on full songs, because hubs also exist on individual MFCC frames, i.e. that there are specific segments of audio data which are close non-perceptive matches to every other possible frames.

6.5.2 Experiment

We build a database of individual 2048-point hamming-windowed frames of audio data, obtained from the uniform segmentation of a few different songs. The database is made to contain 15,000 frames, so results can be quantitatively compared to the full-song behaviour in the Cuidado database. Each frame is modelled by 20 MFCCs (incl. 0^th order coefficient), which is the feature space used in the best performing full-song measure. A distance measure is implemented using euclidean distance, each dimension being normalized to be between 0 and 1, using the 5% and 95% percentile values. This distance measure was chosen to yield a behaviour similar to MFCCs comparison in GMM probability estimation (euclidean comparison with mean vector, rescaled by variance coefficients in each dimension). We compute the 100 nearest neighbors of each frame in the database, store them, and compute the nb of 100-occurrence of each frame in the database.

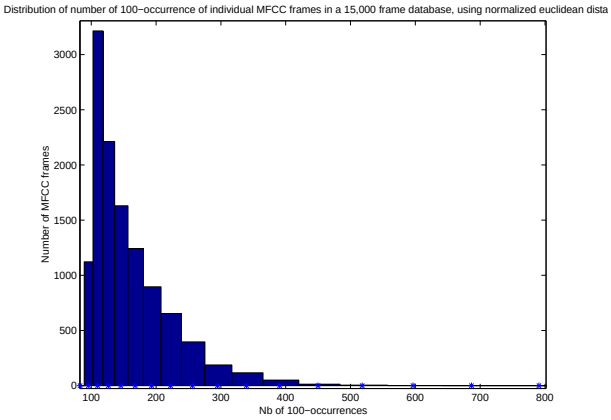

Figure 6.7: Distribution of the MFCC frames according to their number of 100-occurrences in a 15,000-frame database, based on normalized euclidean distance.

6.5.3 Results

Figure 6.7 shows the distribution of the MFCC frames according to their number of 100-occurrences. The distribution is exponentially decreasing, with a maximum N_{100} value around 500. Such small numbers do not indicate the presence of hubs, which is confirmed by manual inspection of the neighbors of the most re-occurring frames. These frames typically correspond to sounds that are common to many different songs, such as noise or silence, and thus have more neighbors than more specific frames (harmonic sounds) that tend to be close to frames of the same song only. The maximum N_{100} value of 500 is less than 10 times smaller than the maximum value obtained for full songs in the Cuidado database. This indicates that the hub phenomenon is not a direct consequence of poor featural representation, but rather an effect of the modelling of the agglomeration of the very many frames in full songs.

6.6 Experiment 4: Influence of modelling

6.6.1 Hypothesis

In this section, we investigate whether hubs are a consequence of a specific algorithmic strategy for modelling the agglomeration of frames in full songs. We test the hypothesis that hubs appear only

(or in majority) for a given algorithm.

6.6.2 Experiment

We compare several measures of hubness on our test database for a subset of the algorithms studied in Chapter 5, chosen to be representative of the principal modelling strategies, namely:

- Static parametric model: 20 MFCCs (incl. 0^{th} coefficient), 50-state GMM, compared by Monte Carlo. This is the best performing algorithm found in Chapter 5.
- Static non-parametric model: 20 MFCCs (incl. 0^{th} coefficient), vector-quantized to 200 codebook vectors using LVQ, modelled by histograms compared by euclidean distance (see Appendix B.7.2).
- Static parametric modelling of first-order dynamics: 20 MFCCs (incl. 0^{th} coefficient), appended with 20 delta coefficients, 50-state GMM, compared by Monte Carlo (see Appendix B.5.1).
- Static parametric modelling of second-order dynamics: 20 MFCCs (incl. 0^{th} coefficient), appended with 20 first-order delta coefficients and 20 second-order acceleration coefficients, 50-state GMM, compared by Monte Carlo (see Appendix B.5.1).
- Dynamic modelling with parametric model: 20 MFCCs (incl. 0^{th} coefficient), modelled with 12-state HMM, using 4 Gaussian components per state, compared by Monte Carlo (see Appendix B.5.3).

6.6.3 Results

Figure 6.8 shows the distribution of the number of 100-occurrences of songs in the test database, for the 5 algorithmic variants. Since the number of occurrences is a constant-sum measure, all 5 distributions are centered on the same mean value of 100 (see Section 6.3.1). However, it appears that the choice of the algorithm has an influence on the shape of the distribution of occurrences. While

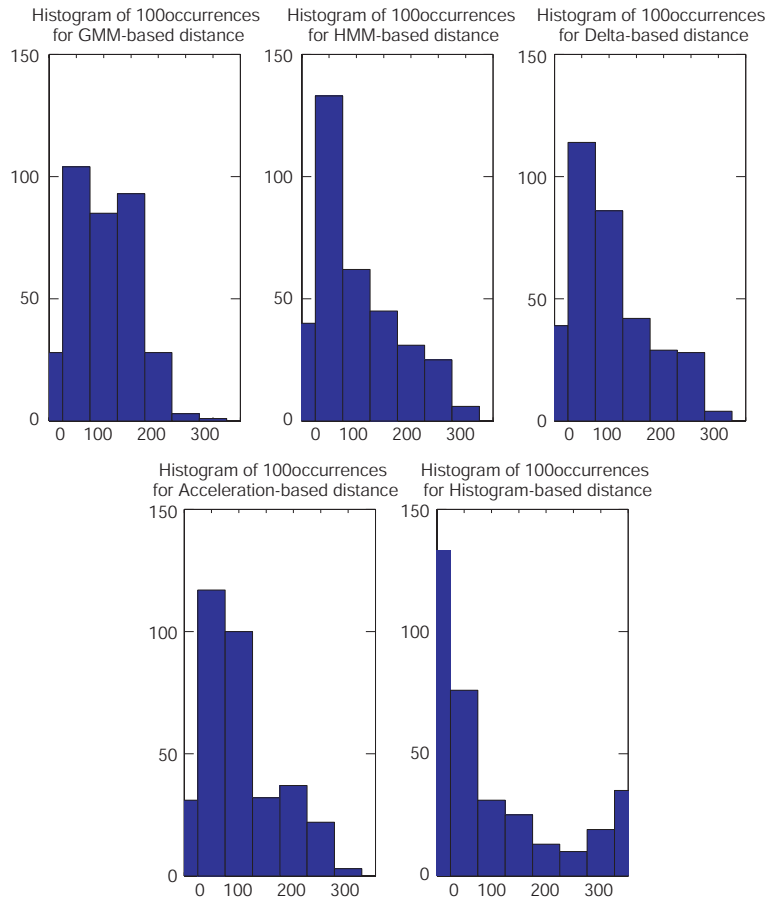

Figure 6.8: Distribution of the Number of 100-occurrences of songs in the test database for several distance algorithms.

all algorithms produce extreme hubs having high number of occurrences (e.g. $N_{100} > 300$), hubs tend to be smaller for the GMM-based distance than for both the dynamic-based and the histogram-based ones. Due to the constant-sum effect, algorithms that produce more high-occurrence songs also produce more low-occurrence songs. This results in a skewed distribution (where very many low-occurrence songs compensate a few high-occurrence songs) in the case of the dynamic-based distances, and a bi-modal distribution for the histogram-based distance, for which very few songs actually take the mean occurrence value.

This behaviour is confirmed by Table 6.4, which shows the number of songs in the test database that exhibit high values for both number of 100-occurrences and number of 20-occurrences. The

5 measures exhibit different proportions of hubs: GMM-based distances produce the fewest, while Histogram-based distance produce 5 times as many.

The proportion of hubs produced by each algorithm is in agreement with the precision reported in Chapter 5: GMM-based distances perform better than (or equivalent to) dynamics, which perform better than histograms.

Nevertheless, it is difficult to conclude that hubs are a specific property of a given algorithmic strategy to model the MFCC frames. All algorithms create hubs. Moreover, static modelling create more hubs than dynamics in the case of Histograms and HMMs, but not in the case of GMM and HMMs. If anything, it seems that non-parametric (Histograms) create more hubs that parametric approaches (GMMs, HMMs). This notably rules out possible convergence problems of parametric estimation (local minima) as a source of bugs.

Table 6.4: Comparison of number of songs exhibiting high number of occurrences in the test database, for several distance algorithms

Measure	GMM	HMM	Delta	Acceleration	Histogram
$N_{100} > 200$	16	48	49	45	69
$N_{20} > 40$	34	41	39	39	42

6.7 Experiment 5: Intrinsic or extrinsic to songs ?

6.7.1 Hypothesis

In this section, we investigate whether hubs are an intrinsic property of given songs, which will act as hubs independently of the algorithm used to model them. We test the hypothesis that hub songs are strongly correlated between different algorithmic measures.

6.7.2 Experiment

We compute the correlation between hubness measures for songs modelled with the same five algorithms as above.

6.7.3 Results

Table 6.5 reports the correlation of the hubness of all songs between various algorithmic models, using 2 measures of hubness (number of 100-occurrences and the neighbor angle).

Table 6.5: Correlation of the hubness of all songs between various algorithmic models. The hubness of songs is measured both by the number of 100-occurrences and the neighbor angle (the latter in parenthesis).

	GMM	HMM	Delta	Acceleration	Histogram
GMM	1.0	0.78 (0.67)	0.79 (0.69)	0.79 (0.71)	0.42 (0.17)
HMM	-	1.0	0.95 (0.96)	0.90 (0.96)	0.47 (0.17)
Delta	-	-	1.0	0.97 (0.99)	0.46 (0.15)
Acceleration	-	-	-	1.0	0.43 (0.14)
Histogram	-	-	-	-	1.0

Both measures reveal the same structure:

- Hubs appearing with GMMs are moderately correlated to HMMs, Delta and Acceleration. This is illustrated on the scatter plot shown in Figure 6.9
- Hubs appearing with HMMs, Delta and Acceleration are very strongly correlated. This is illustrated on the scatter plot shown in Figure 6.10
- Hubs appearing with Histograms are strongly decorrelated to those appearing with the other algorithms.

In more details, Tables 6.6 and 6.7 compare the most frequent hubs for 2 GMM and Histogram-based distances, here measured with their number of 20-occurrences. It appears that some songs act as hubs for both measures, e.g. `MITCHELL, Joni - Dom Juan's Reckless Daughter`. However, a vast majority of the hubs are different. Notably, certain songs are important hubs for one measure and perfectly standard songs for the other. For instance, `SUGAR RAY - Fly` is a hub for the GMM-based distance, but not for the one based on Histograms. Similarly, `CABREL, Francis - Samedi soir sur la Terre` is only a hub for the histogram distance.

We therefore can conclude that:

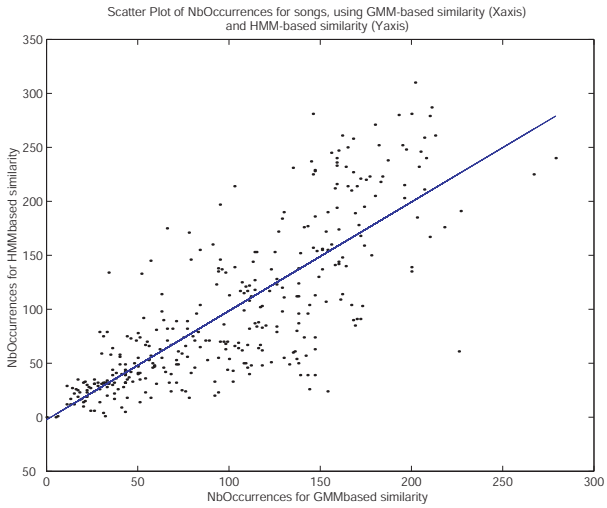

Figure 6.9: Scatter plot of the number of occurrences for songs using GMM-based distance against HMM-based distance.

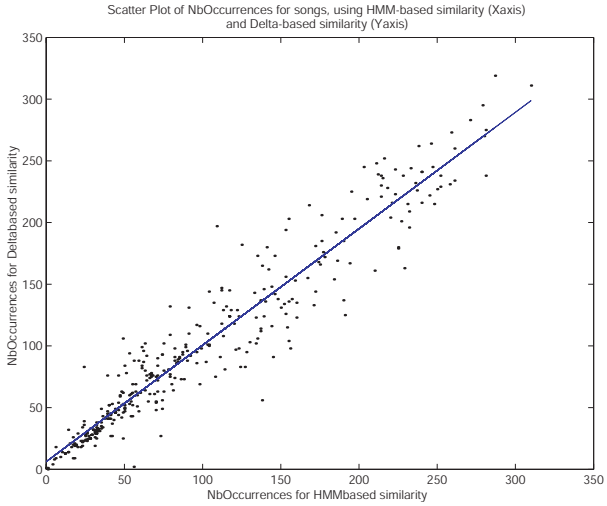

Figure 6.10: Scatter plot of the number of occurrences for songs using HMM-based distance against Delta-based distance.

Table 6.6: Most Frequent False Positives for parametric approach with GMMs

Hubs with {MFCC,GMM}	N_{20} ($card(C_S)$)
MITCHELL, Joni - Don Juan's Reckless Daughter	98(9)
BRIDGEWATER, DD - What A Little Moonlight Can do	79(12)
RASTA BIGOUD - La tchatche est bonne	79(7)
MOORE, Gary - Separate Ways	77(9)
SUGAR RAY - Fly	75(13)
...	
CABREL, Francis - Samedi soir sur la Terre	29 (7)

Table 6.7: Most Frequent False Positives for non parametric approach with Histograms.

Hubs with {VQ,CM}	$N_{20}(card(C_S))$
VOCAL SAMPLING - Radio Reloj	153 (13)
MOORE, Gary - The Hurt inside	126 (9)
CABREL, Francis - Samedi soir su la Terre	122 (7)
CABREL, Francis - Corrida	105
MITCHELL, Joni - Dom Juan's Reckless Daughter	95 (9)
...	
SUGAR RAY - Fly	23(13)

- The hubness of a given song is not an intrinsic property of the song, but rather a property of a given algorithm.

- Dynamics, both via static modelling of dynamical features (delta, acceleration) or via dynamic modelling (HMMs) seems to have an influence of the songs that act as hubs. All three algorithms tend to create the same hubs.

- Parametric modelling tend to create very distinct hubs from non-parametric modelling, so the dynamical/static aspect is not the only involved factor in the appearance of hubs

6.8 Experiment 6: The seductive, but probably wrong, hypothesis of equivalence classes

6.8.1 Hypothesis

An hypothetical explanation for the appearance of hubs can be formulated in terms of *equivalence classes*. The model of the distribution of the feature vectors of a given song, obtained with a given algorithm (e.g. GMM of MFCCs), can generally be obtained identically from several different original datasets, since the transformations of feature extraction and distribution modelling are not bijective operations. Such datasets are said to be equivalent (or invariant) for a given algorithm, and to belong to the equivalence class of the algorithm.

First, typical feature transformations have invariants, i.e. will yield the same results for different signals. For instance, the MFCC algorithm has two notable invariants, due to its use of the Fourier transform:

- MFCC($-$x) = MFCC(x), since FFT($-$x) = FFT(x)
- MFCC(flip(x)) = MFCC(x), since FFT(flip(x)) = $\overline{\text{FFT(x)}}$ (complex conjugate) and thus |FFT(flip(x))| = |FFT(x)|.

While the first transformation doesn't affect the perceptual audio similarity (except for discontinuities at frame transitions), the second (playing a frame backwards) has a strong effect on timbre, notably disrupting the structural succession of transients and steady states.

Second, the models themselves also have invariants. For instance, the GMM models of all permutations of a given set of MFCC frames are identical, since the model doesn't preserve the ordering of the data. Similarly, the HMM models of all permutations *within each HMM state* of MFCCs frames are identical, since this preserves the global markov transition probability matrix. By definition, all equivalent datasets for a given model will be perfect matches to one another. However, model equivalence isn't a sufficient condition for audio similarity: the operation of frame permutation, for instance, severely transforms an audio signal.

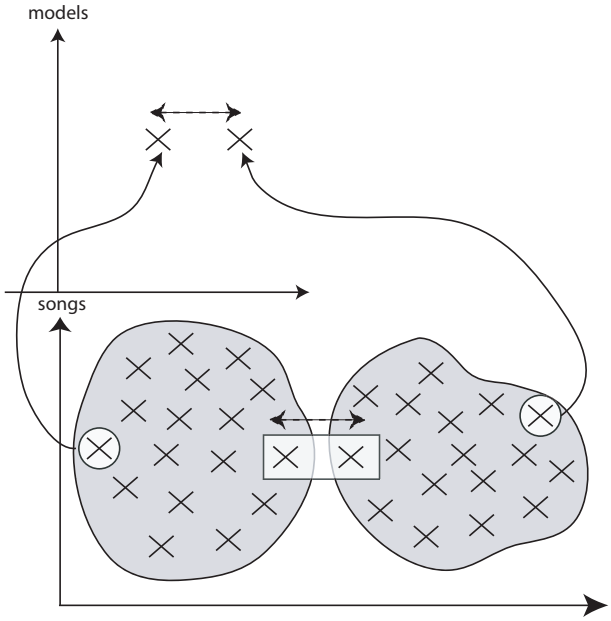

Figure 6.11: Classes of equivalence defined by static models, such that all permutations of the frames of the original song yields the same model

Invariants both at the level of features and models obviously combine: for instance, a signal played backwards will result in a permutation (reverse order) of frames, each corresponding to a flipped signal window, and thus yields the same model than the original when using GMMs of MFCCs. The class of equivalence of a given algorithm can be extremely large. For instance, there are 8000! frame permutations of a typical 3-minute song, and $(2 * 8000)!$ permutations when including the possibility of flipping individual frames.

This model would explain individual bugs. While many signals found in the equivalence class of a given algorithm will probably sound similar, it may be possible to find a duplet of signals which sound very different from one another, and yet have identical models. This is made even more possible when generalizing the equivalence relation ("having the same model") to a pseudo-equivalence ("having similar models"): the models of 2 very different songs may be close because there is a duplet of signals, each equivalent to one of the original songs, which sound similar to one another. This is illustrated in Figure 6.11. In this context also, a hub for a given algorithm is a song for which the equivalence class of the associated model contains signals that are close to signals in

the equivalence class of the models of very many other songs in the database.

6.8.2 Experiment

The hypothesis of equivalence classes has 2 necessary conditions, which can be tested experimentally. According to the model of equivalence classes, a song is more likely to be a hub if the equivalence class of its model given the chosen distance algorithm is large. This will happen notably

- if the song's model, e.g. GMM, has a large variance

- if the modelling algorithm is permissive[‡], which is the case of static algorithms (which allow frame permutations) compared to dynamical models which constraint the permutation to respect the frame-to-frame dynamics of the original frame sequence.

Evidence of the first behaviour can be tested by computing the correlation of some measure of the "diameter" of the GMM model of each song and its degree of hubness, measured e.g. by its number of occurrences. The "diameter" of a GMM model can be defined by sampling a large number of points from this model, measuring the standard deviation of these points in each dimension, and summing the deviations together. This is equivalent to measuring the norm of the covariance matrix of a single-component GMM fitted to the distribution of points. Note that a precise measure of the width of a GMM is non-trivial to compute from the variance of its individual components (e.g. summing them, weighted with each component's prior probability), because this would have to account for the possible overlap between individual components (i.e. computing the volume of the intersection between a set of many ellipsoids in a high dimension space).

Evidence of the second behaviour can be looked for by comparing the global proportion of hubs appearing for static and dynamic algorithms, which was already done in Section 6.6.

[‡]Note that the complexity of a model, in the framework of supervised learning, has received a rigorous mathematical formulation in terms of Vapnik-Chervonenkis dimension, which we do not address here (Blumer et al., 1989)

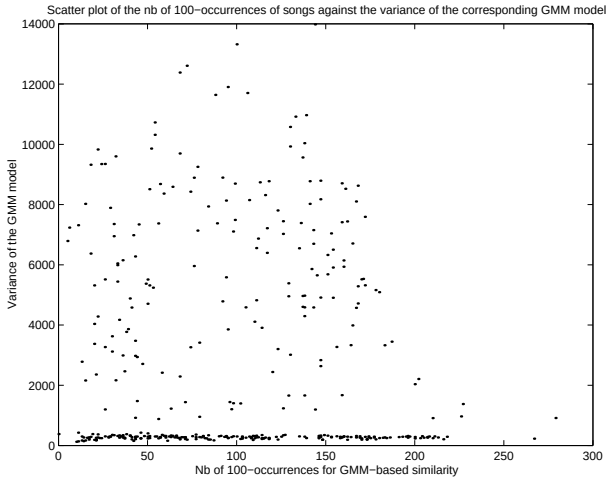

Figure 6.12: Scatter plot of the number of 100-occurrences of songs against the "equivalent variance" of their GMM model, showing no particular correlation.

6.8.3 Results

Figure 6.12 shows a scatter plot of the number of 100-occurrences of songs against the "equivalent standard deviation" of their GMM model. This shows that around 70% of the GMMs' variance lie around the value 300, independantly of the hubness of the song. The remaining GMMs with higher variance neither show any particular correlation with their song's number of occurrence.

Results reported in Section 6.6 show that although static histograms tend to create more hubs than dynamical models such as HMMs, GMM modelling produces as low as half as many hubs than its dynamic counterparts.

Both results strongly disprove the hypothesis of equivalent classes, according to which hubs are explained by unconstrained static algorithms or atypical songs with great variance.

6.9 Experiment 7: On homogeneity

6.9.1 Hypothesis

This section investigates whether the hubness of a given song is a emerging global property of the distribution of its frames, or rather can be localised e.g. to certain frames that are less discriminant

then others.

We have already shown that MFCC frames intrinsically don't exhibit hub behaviours, i.e. one cannot find a specific frame of audio which is close to any other frame, in an euclidean framework. However, this doesn't make any statement about the discriminative power of MFCC frames: it is well possible that most MFCC frames be globally close to one another. This has notably been observed in the domain of speech sounds in Kinnunen et al. (2001). It is therefore possible to imagine that a large part of the distribution of MFCCs is composed of non-discriminative frames, and that what is perceptually salient for a human listener may not be statistically predominant when comparing models of the frame distribution.

6.9.2 Experiment

We describe here an experiment to assess whether there exists such a small portion of the frame distribution which is responsible in majority for the discrimination between non-perceptually close songs. We propose to explore the distribution of MFCC frames by ranking them by statistical importance. In the case of a GMM, frames are all the more so likely to be generated by a given gaussian component c than the weight w_c of the component is high (w_c is also called prior probability of the component).

We define an *homogeneity* transform $h_k : \mathcal{G} \mapsto \mathcal{G}$ on the space $\mathcal{G}$ of all GMMs, where $k \in [0, 1]$ is a percentage value, as:

$g_2 = h_k(g_1)$

$(c_1, c_2, \ldots, c_n) \leftarrow$ `sort(components of` g_1`, decreasing weight)`

`define` $\mathcal{S}(i) = \sum_{j=1}^{i} weight(c_j)$

$i_k \leftarrow \arg\min_{i \in [1,n]} \{\mathcal{S}(i) \geq k\}$

$g_2 \leftarrow$ `newGMM(`i_k`)`

`define` d_i `=component(`g_2`,i)`

$d_i \leftarrow c_i, \ \forall i \in [1, i_k]$

`weight(`d_i`)` $\leftarrow$ `weight(`c_i`)`$/\mathcal{S}(i_k), \forall i \in [1, i_k]$

`return` g_2

end h_k

From a GMM g trained on the total amount of frames of a given song, the transform h_k derives an homogenized version of g which only contains its top k% components. The homogenized GMM accounts for only a subset of the original song's frames: those that amount to the k% most important statistical weight. For instance, $h_{99\%}(g)$ creates a GMM which doesn't account for the 1% least representative frames in the original song.

We apply 11 transforms h_k for $k \in [20, 40, 60, 80, 90, 92, 94, 96, 98, 99, 100]$ to the GMMs corresponding to the optimal measure found in Chapter 5. Each transform is applied to the whole database, thus yielding 11 similarity measures, the properties of which we study below.

6.9.3 Results

Variance

Figure 6.13 shows the influence of the homogenization transform on the variance of the resulting GMM. The variance of the model is evaluated with the sampling procedure already described in Section 6.8. It appears that the homogenization transform reduces the variance of the models exponentially. This suggests that the least representative points, which are removed in the last 5-10% of the total distribution, account for most of the variance of the global distribution, and probably represent very different MFCC frames than the ones composing the main mass of the distribution.

Influence on hubs

Figure 6.14 and 6.15 show the influence of the homogenization transform on the number of hubs in the database. Hubness is measured in the case of Figure 6.14 by the number of songs in the test database having a number of 100-occurrences greater than 200, and in the case of Figure 6.15, by the number of songs with a mean neighbor angle greater than 65°.

Both metrics indicate that GMM homogenization critically increases the number of big hubs

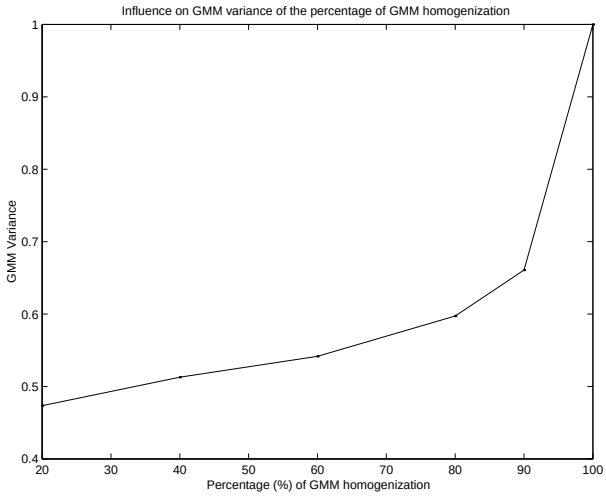

Figure 6.13: Influence of the percentage of GMM homogenization on the variance of the resulting GMM.

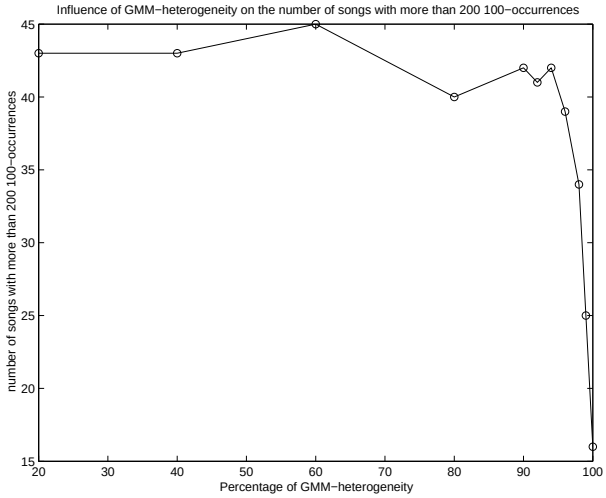

Figure 6.14: Influence of the percentage of GMM homogenization on the number of songs with more than 200 100-occurrences

in the database: homogenization with $k = 30\%$ creates more than twice as many hubs with more than 200 occurrences, and more than 5 times as many hubs with angles greater than 65°. As homogenization reduces the variance of the models, this gives further experimental disproval of the hypothesis examined above in Section 6.8. It seems reasonable to interpret the increase of hubness when k decreases as a consequence of reducing the amount of discriminative information in the GMMs (i.e. from representing a given song, down to a more global style of music, down to the even simpler fact that it *is* music). This is the same to say that all songs share the same most important 20% of frames.

However, the remaining 80% do not monotonically increase the discrimination of a given song from its neighbors. Both figures clearly show a very important increase in the number of hubs in the first few percent of homogenization. The extreme number of hubs obtained with $k = 30\%$ is reached as early as $k = 92\%$ in the case of the occurrence metric and $k = 96\%$ in the case of the mean angle metric. This is a strong observation: the hubness (or rather non-hubness) of a song seems to be controlled by an extremelly small amount of critical frames, which represent typically less than 5% of whole distribution. Moreover, these frames are the least statistically significant ones, i.e. are modelled by the least important gaussian components in the GMMs. This indicates that the majority (more than 90%) of the MFCC frames of a given song are a very poor representation of what discriminates this song from other songs.

Moreover, Figure 6.15 shows that after the extremely rapid peak of hubs when removing the first 5% frames, the number of hub songs tend to decrease when k decreases from 90% to 60%, and then increases again for k smaller than 60%. The minimum value reached at $k = 60\%$ is equivalent to the original value at $k = 100\%$. A similar decreasing behaviour is observable to a smaller extent with the other metric in Figure 6.14 (with a local minimum at $k = 80\%$), although it is difficult to establish that this is a statistically significant trend.

The behaviour in Figure 6.15 suggests that there is a population of frames in the range $[60\%, 95\%]$ which is mainly responsible for the hub behaviour. While the hubness of songs diminishes as more frames are included when k increases from 20% to 60% (such frames are increasingly

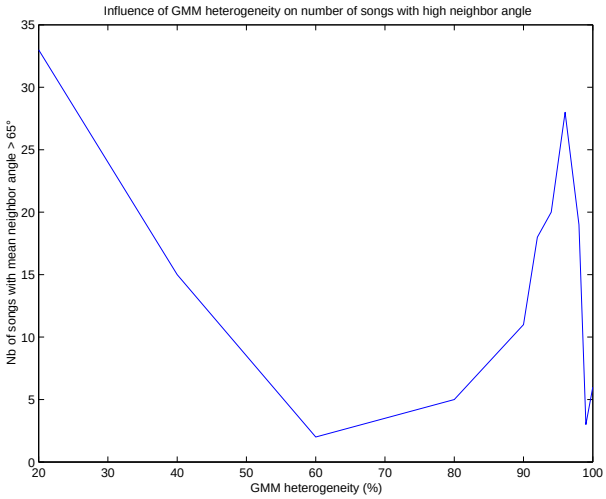

Figure 6.15: Influence of the percentage of GMM homogenization on the number of songs with a mean neighbor angle greater than 65°

specific to the song being modelled), it suddenly increases when k gets higher than 60%, i.e. this new 30% information is detrimental for the modelling and tend to diminish the discrimination between songs. The continuous degradation from 60% to 95% is only eventually compensated by the inclusion of the final 5% critical frames.

Influence on precision

Figure 6.16 shows the influence of homogenization on the *R*-precision of the resulting similarity measure. The figure closely mimics the (inverse) behaviour seen in Figure 6.15, with precision plummeting when k decreases from 100% to 92%, and then reaching a local maximum again between 60% and 80%. This gives further support to the observation that not all frames are equally discriminative, and that there exists a population of frames in the range $[60\%, 95\%]$ which is detrimental to the modelling of perceptual similarity.

The influence of non-discriminative frames indicates that timbre models would probably benefit from some kind of discriminative training (Ulusoy and Bishop, 2005), where frames that don't help comparing songs to one another are typically not modelled in individual songs. However, the framework of supervised training is not particularly well suited to similarity tasks, in which we

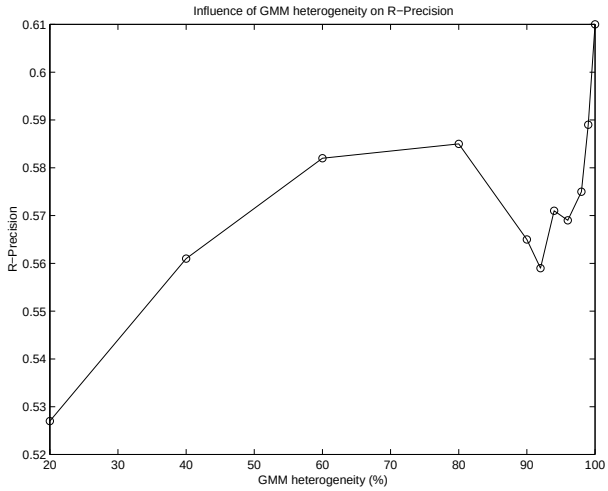

Figure 6.16: Influence of the percentage of GMM homogenization on the *R*-precision of the similarity measure

don't only want to discriminate but also to compare, and thus should be adapted to our situation. Appendix B.7.2 illustrates such a case where the direct application of supervised discriminative algorithms (namely LVQ) does not provide enough basis for comparing different songs accurately.

6.10 Experiment 8: Are hubs a structural property of the algorithms ?

6.10.1 Hypothesis

Section 6.7 establishes that hubness is not an intrinsic property of a given song, but rather is dependent on the modelling algorithm. In this section, we investigate whether hubs are a structural property of pattern recognition-based similarity measures, and that they can be observed in any dataset. This is a relevant question knowing as remarked earlier that hubs have been observed in this study on timbre similarity, but also in the domain of Speaker and Fingerprint identification.

6.10.2 Experiment

We apply the same modelling technique (GMMs of MFCCs) to compute the perceptual similarity of another class of audio signals, namely ecological sound textures. We gathered a database of 106

3-minute urban sound ambiances, recorded in Paris using a omni-directional microphone [§]. The recordings are clustered in 4 “general classes”:

- Boulevard: Recordings made on relatively busy boulevards, with predominant traffic noise, notably buses and car horns.
- Neighborhood: Recordings made on calmer neighborhood streets, with more diffuse traffic, notably motorcycles, and pedestrian sounds.
- Street Market: Recordings made on street markets in activity, with distant traffic noise and predominant pedestrian sounds, conversation and auction shouts.
- Park: Recordings made in urban parks, with lower overall energy level, distant and diffuse traffic noises, and predominant nature sounds, such as water or bird songs.

Recordings are further labeled into 11 “detailed classes”, which correspond to the place and date of recording of a given environment. For instance, “Parc Montsouris, Paris 14e” is a subclass of the general “Park” class. Some detailed classes also discriminate takes at identical places and dates, but with some exceptional salient difference. For instance, “Marché Richard Lenoir, Paris 11e” is a recordings made in a street market on Boulevard Richard Lenoir in Paris, and “Marché Richard Lenoir (music)” is a recording made on the same day of the same environment, only with the additional sound of a music band playing in the street. Table 6.8 shows the details of the classes used, and the number of recordings available in each class.

Each audio recording is modelled with 50-ms frames, 20-MFCCs and 50-state GMMs. Models are compared to one another with Monte-Carlo distance using 2000 samples. We measure the precision of measure using the same framework as for music timbre similarity, i.e. by counting the number of recordings having the same class as the seed item. We report on the properties of the resulting similarity function below.

[§]This material was collected and kindly made available by Boris Defreville from LASA.

Table 6.8: Composition of the ecological sound database.

Class	Detailed Class	Number of instances
Boulevard	Boulevard Arago	14
Boulevard	Boulevard du Trone	5
Boulevard	Boulevard des Maréchaux	8
Street	Rue de la Santé (14e)	7
Street	Rue Reille day1 (14e)	14
Street	Rue Reille day2 (14e)	7
Market	Marché Glaciére	8
Market	Marché Richard Lenoir	22
Market	Marché Richard Lenoir (music)	9
Park	Parc Montsouris Spring	20
Park	Parc Montsouris Summer	8

6.10.3 Results

Precision

Table 6.9 gives the precision of timbre similarity applied to ecological sound textures. It appears that the results are substantially better than for polyphonic music signals, nearing perfect precision in the first 5 nearest neighbors even for detailed classes. High precision using the general classes shows that the algorithm is able to match recordings of different places on the basis of their sound level (boulevards, streets), and sound quality (pedestrian, birds). High precision on detailed classes shows that the algorithm is also able to distinguish recordings of the same environment made at different times (Spring or Summer), or in different contexts (with and without music band). This result has a natural application to the classification of ecological recordings, e.g. using a simple k-nearest neighbor strategy, and could prove useful for context-recognition, for instance in the context of wearable computing (Clarkson et al., 2000).

Table 6.9: Precision of timbre similarity applied to ecological sound textures.

Ground Truth	5-Precision	10-Precision	15-Precision	*R*-Precision
General Class	0.94	0.87	0.77	0.66
Detailed Class	0.90	0.79	0.75	0.74

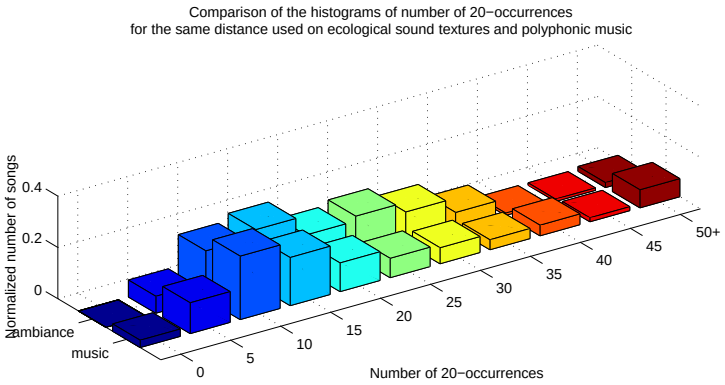

Figure 6.17: Comparison of the histograms of number of 20-occurrences for the same distance used on ecological sound ambiances and polyphonic music.

Hubs

Figure 6.17 shows the histogram of the number of 20-occurrences obtained with the above distance on the database of ecological sound ambiances, compared with the same measure on the test database of polyphonic music. It appears that the distribution of number of occurrences for ambiance sounds is more narrow around the mean value of 20, and has a smaller tail than the distribution for polyphonic music. Notably, there are four times as many audio items with more than 40 20-occurrences in the music dataset than in the ambiance dataset. This is also confirmed by the manual examination of the similarity results for the ecological ambiances: none of the (few) false positives re-occur significantly more than random.

This establishes the fact that hubs are not an intrinsic property of the class of algorithm used here, but rather appear only for a certain classes of signals, among whom polyphonic music, but not ecological sound ambiances.

Homogeneity

Ecological sound ambiances and polyphonic music are two different datasets for which hubs appear in the second case, but not in the first. As we will see here, the two classes of signals can notably be distinguished in terms of their homogeneity.

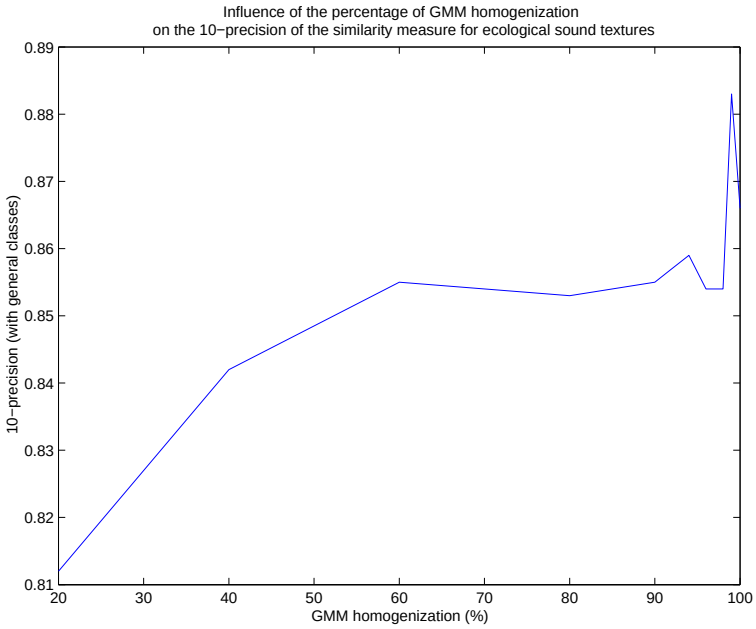

Figure 6.18: Influence of the percentage of GMM homogenization on the 10-precision (with general classes) of the similarity measure used on ecological sound ambiances.

Figure 6.18 shows the influence of the homogeneity transform introduced in Section 6.9.2 on the precision of the similarity measure applied to ecological sound ambiances. We notice a very different behaviour than for polyphonic music, for which the filtering of the very first few frames had a dramatic impact on the precision. In the case of sound ambiances, 99% homogenization is slightly beneficial to the precision. This suggests that the 1% less significant frames are spurious frames which are worth smoothing out. Further homogenization down to 60% has a moderate impact on the precision, which is reduced by about 1% (absolute). This suggests that the frame distribution is very homogeneous, and doesn't exhibit critical populations of frames which are either extremely discriminative (such as the $[95\%, 100\%]$ region for polyphonic music), or non-discriminative (such as the $[60\%, 95\%]$ region for polyphonic music). Ecological ambiances can be discriminated nearly optimally by considering only the most significant 50% of the frames.

To further assess the different homogeneity between both datasets, we consider an alternative data transformation, *n-folding*, which folds an original signal onto itself a number of times (as seen

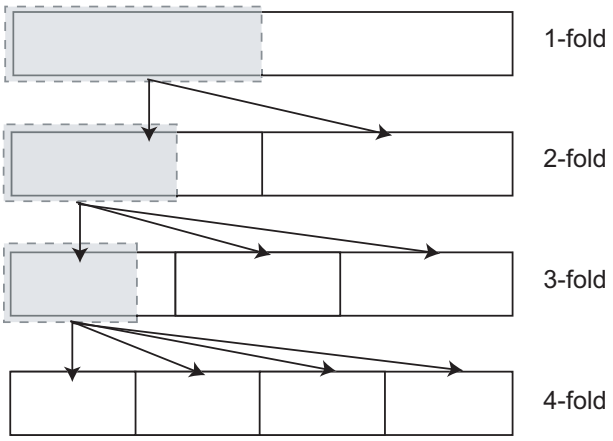

Figure 6.19: The n-fold transform creates increasingly homogeneous signals by folding a reduced portion of the original signal.

in Figure 6.19). The output of the 2-fold transform is 50%-sized random extract from the original, repeated twice. Similarly, the 3-fold transform is a 33%-sized extract of the original repeated three times. All signals processed by n-folding from a given signal have the same duration as the original, but contain less "varied" material. Note that since the duration of the fold (an integer division of the total duration) is not a multiple of the frame duration in the general case, n-folding doesn't simply duplicates the MFCC frames of the folded extract, but rather creates some limited jitter. The fact that all n-folded signals have the same number of frames as the original enables to use the same modelling parameters, notably number of gaussian components (else we would have had to account for the curse of dimensionality).

We apply n-folding to both datasets (ambiances and music), for $n \in [1, 2, 3, 4, 5, 10, 20, 30, 50]$. Figure 6.20 shows the influence of folding on the similarity R-precision for both classes of signals (where both precision curves are normalized with respect to their maximum). N-folding is detrimental to the precision for both datasets. However, it appears that ecological sound textures are typically twice more robust to folding than polyphonic signals. Considering only a tenth of the audio signals cuts down precision by 15% for sound textures, and by more than 35% for polyphonic music. In the extreme case of folding only 3 seconds out of a 3-minutes sound extract (50-folding), the precision loss is 20% for ecological sounds, but more than 60% for polyphonic music.

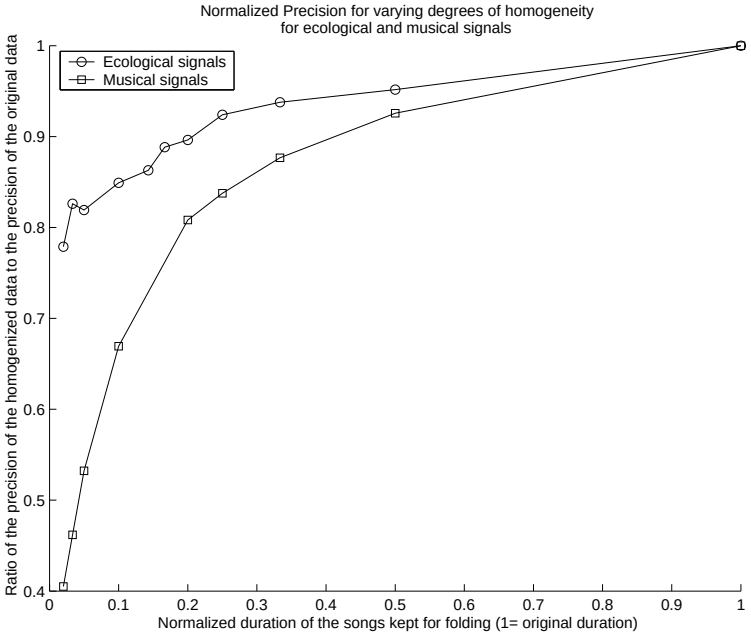

Figure 6.20: Comparison of the influence of k-folding on the precision of the same distance algorithm used on ecological sound ambiances and polyphonic music.

This suggests that frame-based feature distributions for ecological sound textures are statistically much more self-similar than polyphonic music, i.e. they can be compressed to only a small fraction of their duration without much loss in terms of distribution modelling. If we authorize a 10% precision loss, ecological signals can be reduced to around 10-second extract. Polyphonic music on the contrary seems to require a large quantity of feature information in order to be properly modelled and compared: the same 10% tolerance requires more than 1 minute of data. Note that the former is comparable to the human performance measured in Peltonen et al. (2001) on the task of recognizing everyday auditory scenes (20 seconds). However, the latter (polyphonic music) is many times less effective than humans (Perrot, 1999).

The greater heterogeneity of polyphonic music data for pattern recognition purpose may explain the appearance of hubs, and their non-existence for other, more homogeneous classes of signals. It would be worth investigating the feature-homogeneity of other hub-prone classes of signals, such as speaker data or fingerprints, to give further support to this hypothesis.

Chapter 7

Experiments 9 & 10: Grounding

The two experiments which conclude this study make a round trip back to high-level music descriptors. In Chapters 5 to 6, we explicitly evaluated the validity of polyphonic timbre models based on the "GMM of MFCCs" approach described in Chapter 4. We now examine the validity of using such timbre models to extract information such as musical genre. We base our study on a yet-unreleased very large and diverse set of manually collected metadata, made available to Sony CSL by collaborations with the Sony Corporation. We show in **Experiment 9** that surprisingly few high-level descriptors are directly correlated to timbre. Moreover, different taxons of a given category, such as "Mood Violent" or "Mood Ironical", have very diverse levels of correlation with timbre (high and low resp.), which is at odds with typically proposals of classifiers that apply the same decision space for every taxons. However, **Experiment 10** shows that there are extreme amounts of correlation between high-level descriptors, independently of their relation to timbre. Some of these correlations capture psycholinguistical semantic associations ("a powerful song is a strong song"), but also historical and cultural knowledge ("rock uses guitars"), and more subjective aspects linked to perception of timbre ("flute sounds warm"). This suggests that very many high-level cultural descriptions of music *can* indeed be grounded to timbre similarity, by exploiting such higher-level correlations with timbre-based attributes. We propose a hybrid classification system that implements this idea in a systematic way.

7.1 Experiment 9: Inferring high-level descriptions with timbre similarity

This experiment examines the validity of using models of polyphonic timbre similarity to extract high-level music descriptions. As reviewed in Chapter 2, the assumption that concepts like musical genre or mood are grounded on the global sound of a given piece of music is common to more than a fourth of all contributions in the history of MIR research. This assumption is notably motivated by the difficulty to address other factors such as harmony or rhythm, but also and foremost cultural interpretations, which depend on epochs, locations and user communities. This experiment evaluates the precision of an inference mechanism based on the best timbre similarity measure obtained in this work, on a very large and diverse set of manually collected metadata.

7.1.1 Material

We base our study on a yet-unreleased large set of editorial metadata values made available to Sony CSL by partnerships within the Sony Corporation. The database currently contains 4936 songs, each described by a set of 801 boolean attributes (e.g. "Language English"= *true*). These attributes are grouped in 18 categories, which can be found in Table 7.1.

Attribute values were filled in manually by a specialised subcontractor. The high-level descriptions found in the database are very diverse. Some attributes are correlated with some acoustic aspect of the sound ("Main Instrument","Dynamics"), while others seem to result from a more cultural view on the music object ("Genre", "Mood", "Situation"). Nearly each of these broad categories have seen prior attempts at automatic classification, although often with much smaller taxonomies. Liu et al. (2003) for instance propose a system to classify the mood of a song into 4 taxons: *Contentment*,*Depression*, *Exuberance* and *Anxious*. One can see from Table 7.1 that the database we consider here offers 58 of such mood taxons. Hence, this database is a realistic and quite thorough coverage of the diversity of descriptors considered in typical MIR systems.

One should note that category taxonomies are not intended for universality: the definition of

Table 7.1: Categories of the attributes used in the database

Category	Nb attributes	Example attribute
Aera/Epoch	16	1970-1980
Affiliate	5	Germany
Character	39	Child-oriented
Country	31	Brazil
Dynamics	4	Decreasing
Genre	36	Jazz Standard
Language	15	Spanish
Main Instrument	107	Contra Bass (pizz.)
Metric	14	3/4
Mood	58	Aggressive
Musical Setup	25	String Ensemble
Rhythmics	10	Groovy
Situation	82	City By Night
Special Creative Period	3	Early
Style	176	Bebop
Tempo	8	Slow - Adagio
Text Category	123	Forgiveness
Variant	46	Natural / Acoustic

attributes such as “Style Alternative Rock” and how they differ from, say, “Style Rock Pop” is a convention which is only local to the categorizing company, and to which we didn’t have access in this study. Therefore these attributes are not primarily intended for direct informative display, but rather for creating a mid-level representation which can be used for matching and recommendation. For all these reasons, we propose here to analyse this set of attributes as an arbitrary ontology only defined by the values taken on the database, and not to consider any exterior musical assumption of what a “Genre” or “Style” should be.

7.1.2 Methods

We propose to infer the value of a given attribute $\mathcal{A}$ for a given song $\mathcal{S}$ by looking at the values of $\mathcal{A}$ for songs that are timbrally similar to $\mathcal{S}$. More precisely, we define as our observation O_S the

number of songs among the set $\mathcal{N}_S$ of the 10 nearest neighbors of $\mathcal{S}$ for which $\mathcal{A}$ is *true*, i.e.

$$\mathcal{O}_S = card\{\mathcal{S}_i \setminus \mathcal{S}_i \in \mathcal{N}_S \wedge \mathcal{A}(\mathcal{S}_i)\} \tag{7.1}$$

If the attribute $\mathcal{A}$ is correlated with timbre, large values of $\mathcal{O}_S$ are a good indicator that $\mathcal{A}(\mathcal{S})$ is *true*. For instance, if 9 out of the 10 nearest neighbors of a given song are "Hard Rock" songs, then it is very likely that the seed song be a "Hard Rock" song itself. However, we need to compensate this decision by the fact that attributes are not uniformly distributed in our database subset. For instance, "Genre Dance Music" is *true* for 4123 songs out of 4936, while "Main Instrument Bandoneon" has only one positive example. We thus define $P(\mathcal{A}(\mathcal{S})/\mathcal{O}_S)$ the probability that $\mathcal{A}$ be true for $\mathcal{S}$ given the observation $\mathcal{O}_S$ of a given number of true values in the set of nearest neighbors, and $P(\overline{\mathcal{A}(\mathcal{S})}/\mathcal{O}_S)$ the probability that $\mathcal{A}$ be false given the same observation. According to Bayes' law,

$$p(\mathcal{A}(\mathcal{S})/\mathcal{O}_S) = p(\mathcal{O}_S/\mathcal{A}(\mathcal{S}))\frac{P(\mathcal{A}(\mathcal{S}))}{P(\mathcal{O}_S)} \tag{7.2}$$

The likelihood distribution $p(\mathcal{O}_S/\mathcal{A}(\mathcal{S}))$ can easily be estimated by the histogram of the empirical frequencies of the number of positive neighbors for all songs having $\mathcal{A}(\mathcal{S}) = true$ (similarly for $P(\overline{\mathcal{A}(\mathcal{S})}/\mathcal{O}_S)$). Figure 7.1 shows 2 examples of such likelihood distributions computed for attributes "Character Calm" and "Genre Club/Discotheque". If we assume a flat prior

$$P(\mathcal{A}(\mathcal{S}) = P(\overline{\mathcal{A}(\mathcal{S})} = 0.5 \tag{7.3}$$

we can estimate $\mathcal{A}(\mathcal{S})$ using the maximum likelihood criteria :

$$\mathcal{A}(\mathcal{S}) = p(\mathcal{O}_S/\mathcal{A}(\mathcal{S})) > p(\mathcal{O}_S/\overline{\mathcal{A}(\mathcal{S})}) \tag{7.4}$$

Using the example given in figure 7.1, we see that under the observation that 4 nearest neighbors out of 10 have "Character Calm", we estimate that the seed song has "Character Calm". However,

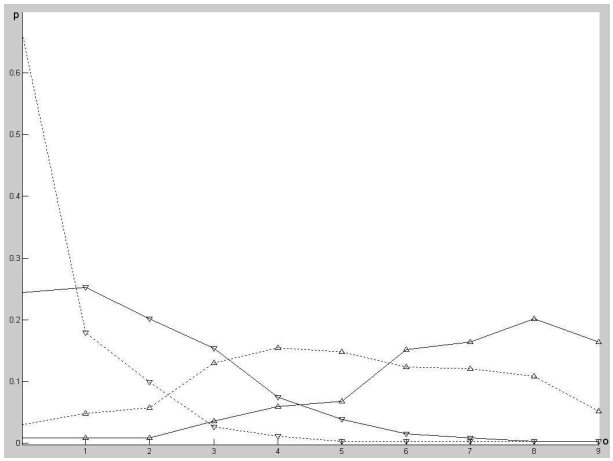

Figure 7.1: Likelihood distributions for 2 attributes “Character Calm” (downward pointing triangle) and “Genre Club/Discotheque” (upward pointing triangle). Positive likelihood $p(O_S/\mathcal{A}(\mathcal{S}) = true)$ appear in solid line, and negative likelihood $p(O_S/\mathcal{A}(\mathcal{S}) = false)$ in dashed line. The x-axis corresponds to the number of songs having $\mathcal{A}(\mathcal{S}) = true$ observed in the first 10 nearest neighbor of the seed song.

under the same observation that 4 nearest neighbors out of 10 have “Genre Club/Discotheque”, we estimate that the seed song does not have “Genre Club/Discotheque”, because this observation is in fact surprisingly small given the large number of songs of “Genre Club/Discotheque” present in the whole database.

7.1.3 Results

Surprisingly few attributes are correlated with timbre

Figure 7.2 shows the distribution of the precision of the timbre inference process described above on the set of the 801 boolean attributes in the database. It appears that some attributes are very correlated to timbre similarity (in the sense defined above), achieving precisions sometimes higher than 95%. However, there are suprisingly few of such near-perfect correspondances. Only 6% of the attributes in the database are estimated with more than 80% precision, and more than a half of the database’s attributes are estimated with less that 65% precision (which hardly better than a binary random choice, i.e. 50%).

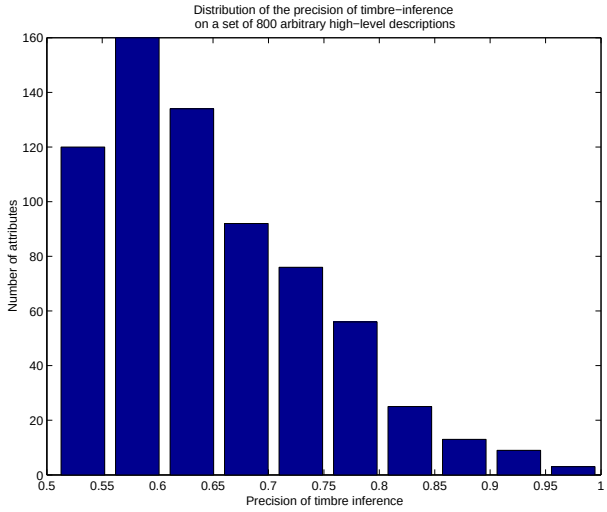

Figure 7.2: Distribution of the precision of timbre inference on the set of 800 attributes.

Table 7.2 shows the 10 attributes (out of 801) that were the most precisely inferred using timbre similarity. Most attributes correspond to "Styles" which described music with quite extreme observable features of the signal (e.g. saturated guitar, distorted vocals, high percussivity).

It should be mentioned that the nearest neighbor algorithm defined above is not a particularly flexible classification algorithm, and notably can have difficulties with strongly multimodal distributions ("Songs of category X sound either like A, or like B, or like C"). However, these results indicate that very few typical high-level music descriptions have a consensual definition in terms of a prototypical "timbre".

Not all taxons of a given category behave similarly

Table 7.3 shows the distribution of the categories of the attributes which are inferred with more than 75% precision, while Table 7.4 shows the distribution of the attributes which are inferred with less than 55% precision.

From these two tables, we observe that many categories include both timbrally-related and unrelated attributes. "Genre unplugged", "Style Hard Rock" (found in Table 7.3) are strong timbre correlates (in the sense defined above), mostly because the instances found in the database are very

Table 7.2: Best inferred attributes. P_+ (resp. P_-) is the ratio of the number of correctly inferred *true* (resp. *false*) values over the total number of *true* (resp.*false*) values.

Attribute	P_+	P_-	Mean Precision
Style Techno (minimal)	0.93	0.99	0.96
Style Rave	0.94	0.98	0.96
Genre Lullaby/Nursery Rhyme	0.91	0.99	0.95
Style Hard Trance	0.92	0.95	0.93
Language Native American	0.88	0.98	0.93
Style Garage	0.87	0.99	0.93
Style Happy Hardcore	0.86	0.98	0.92
Style Metal	0.88	0.96	0.92
Style Hardcore	0.86	0.98	0.92
Style Grunge	0.87	0.95	0.91

Table 7.3: Categories of the best inferred attributes

Category	Nb attributes	Sample Attributes
Style	48	Jazz (Trad.), Hard Rock
Genre	10	Unplugged, Nightclub Music
Main Instrument	10	Guitar (distortion), Vocals (Spoken; Rap)
Aera/Epoch	9	1950-1960, 1960-1970
Musical Setup	7	Big Band, Rock band
Character	6	Metalic, Warm
Country	6	Cuba, Jamaica
Variant	3	Aggressive, Metallic
Situation	3	Computer Animation, Middle Ages
Mood	2	Aggressive, Negative
Language	1	Native American

prototypical, and timbrally consistent (e.g. salient saturated guitar and strong percussions in Hard Rock), while “Genre Jingle” and “Style Electronica” (found in Table 7.4) are poor timbre correlates, possibly because they are very heterogeneous. “Electronica ” for instance spans possibly everything from HardCore Techno - solely percussive -, electronic pop (artists like Emilie Simon or Air) where voice is predominant, Intelligent Techno - which uses concrete sound recordings and electronic blips - and even margin artists like Craig Armstrong which really does symphonic orchestral music.

Table 7.3 and 7.4 also confirm that categories like “TextCategory”, “Situation” or “Mood” mostly capture cultural and subjective information which are poorly described with timbre. Never-

Table 7.4: Categories of the worst inferred attributes

Category	Nb attributes	Sample Attributes
TextCategory	35	Irony, Play on Words
Main Instrument	18	Clarinet, Body Percussion
Style	17	Underground, Electronica
Situation	10	Hollywood, Winter
Variant	6	Thin, Wrong / Amateurish
Mood	6	Maritime, Funny
Country	5	International, USA
Musical Setup	4	Duo, Girlgroup
Character	3	Child-oriented, Vibrating
Metric	2	6/8, 7/8
Dynamics	1	Decreasing
Genre	1	Jingle/Link
Language	1	African Languages
Tempo	1	Alternating

theless, taxons like “TextCategory Explicit” or “Mood Violent” are very good timbre-correlates.

This is at odds with typically proposals of classifiers that apply the same decision space for every taxons of a given category. What appears from these results is that only a few taxons, wide-spread over many diverse categories, can reliably be inferred with timbre.

The paradoxical subjectivity of timbre judgements

It also appears that attributes of the “Main Instrument” category are not particularly well modelled by timbre. This can be explained by the fact that instruments described by such attributes are usually not salient throughout the song, if salient at all. This illustrates the subjectivity of the categorization of music: many of our “timbre” judgements are not low-level immediate perceptions, but rather high-level cognitive reasoning which accounts for the evidence found in the signal, but also depends on cultural expectations, a priori knowledge, interestingness and “remarkability” of an event, etc. A given song by e.g. *Elton John* may be labeled as “piano” music, even though one can barely hear any piano sound on careful inspection, e.g. because it is very distant in a mix with predominant strings and synthetic pads, or because it is heavily processed with audio effects such as flangers and

delays. However, the knowledge that *Elton John* is a pianist, or that this particular song bears some similarity with another piano song (which itself may not have salient piano either...) enables some kind of "automatic completion" of what is perceived onto what is thought to be perceived. Such paradoxes or timbre illusions are proper to the perception of complex polyphonic music, and most notably of "known music" which can be mapped to a space of music pieces a particular listener already knows. (e.g. "Elton John" finding room in my musical universe among other British pop singers, but also other pop pianists like Paolo Conte, etc.)

7.2 Experiment 10: The use of context

In this experiment, we investigate whether there exist statistical dependencies between the attributes in the database, which would reveal the kind of correlations and context-based inference that seems at play with timbre judgements. To do so, we measure the statistical independence between all pairs of attributes in the database.

7.2.1 Method

A common way to assess the statistical independence of pairs of attributes - considered here as random variables- is to use Pearson's χ^2-test (Freedman et al., 1997). This tests the hypothesis (called the *null hypothesis*) that the relative frequencies of occurrence of observed events follow a flat random distribution (e.g. that hard rock songs are not significantly more likely to talk about violence than non hard-rock songs). χ^2 is calculated by finding the difference between observed and theoretical frequency for each attribute. The result is normalized by the size of the population, yielding a value Φ between 0 (corresponding to no association between the variables) and 1 (complete association). Appendix G gives more details about the computation of χ^2 and Φ values.

7.2.2 Results

Tautologies

Tables 7.5 to 7.9 show a classification of the 100 most correlated duplets of attributes on the whole database, assessed using the Φ coefficient. We observe in Table 7.5 that a number of such correlations translate trivial word-to-word associations between attributes, such as "TextCategory Christmas" and "Situation Christmas". These sometimes uncover unexplained redundancies in the attribute taxonomies, such as the co-existence of attributes like "Aera 1980's" and "Aera 1980-1990", but also reveal the existence of logical links between e.g. "Characters" and "Variants" of the same name (calm, metallic, simple, etc.) with a consistency which is remarkable in the context of massive manual categorization. This may suggest that such logical links are enforced by the categorizing company with specially designed GUI or post-processing routines.

Table 7.5: Most correlated duplets of attributes - redundant information

Attribute1	Attribute2	Φ
Language Finish	Country Finland	0.93
Textcategory Christmas	Situation Christmas	0.81
Country Italy	Affiliate Affiliate Italy	0.74
Aera/Epoch 1980-1990	Aera/Epoch 80s	0.72
Genre Live Classic	Genre Live	0.72
Aera/Epoch 1970-1980	Aera/Epoch 70s	0.71
Mood aggressive	Variant aggressive	0.70
Character pulsating	Mood pulsating	0.69
Musical Setup Jazz Band	Style Jazz	0.67
Style traditional Country	Musical Setup Country Band	0.66
Style Jazz (trad.)	Genre Jazz Standard	0.66
Character distorted	Variant Distortion	0.65
Character simple	Variant simple	0.65
Mood nostalgic	Character nostalgic	0.63

Overfitting

Table 7.6 illustrates a caveat which is common to many of the results reported in this chapter, namely the risk of overfitting. Here, the strong correlation between songs that use Jew's Harp instrument

(also called jaw's harp or gewgaw) and the fact that these songs talk about "bicycle" can only be explained by the fact that there exists only one song in the database that exhibits this somewhat odd combination ("Luka Bloom - The Acoustic motorbike"). Similarly, most Spanish songs in the database are "ska punk" songs, and most appearing jazz standards are played with a transverse flute. There are a number of techniques to automatically detect such overfits (notably n-fold cross validation).

Table 7.6: Most correlated duplets of attributes - overfitting

Attribute1	Attribute2	Φ
Main Instruments Jew's Harp	Textcategory Bicycle	1
Affiliate Affiliate Spain	Style Ska Rock	0.73
Country Spain	Style Ska Rock	0.62
Country Spain	Style Punkrock	0.52
Main Instruments transverse flute	Genre Jazz Standard	0.50

Exclusions

Table 7.7 illustrates that such analysis is also able to track down negative correlation, i.e. mutual exclusion relationships. For instance, a single song can't at the same time have varying and steady dynamics, or be both vocal and instrumental.

Table 7.7: Most correlated duplets of attributes - negative correlation

Attribute1	Attribute2	Φ
Dynamics dynamic (up+down)	Dynamics steady	0.80
Main Instruments male	Main Instruments female	0.70
Main Instruments Vocals	Language Instrumental	0.61
Language English	Language Instrumental	0.58

Intrinsic semantics

Table 7.8 shows a number of associations that result from intrinsic semantic correlations, which have little to do with the actual musical usage of the words. For instance, the analysis reveals common-sense relations such as "Christmas" and "Special occasions", "Well-known" and "Popu-

lar", "Strong" and "Powerfull". Associations such as these and the ones exhibited in Table 7.7 are less easy to infer a priori than the word-word associations in Table 7.5. Typically, this would use some kind of lexical reference system where nouns, verbs, adjectives and adverbs are organized into synonym sets (see e.g. WordNet Fellbaum (1998)). This also illustrates that the manual categorization process is consistent with psycholinguistics evidences of semantic associations, and that the specific usage of words that describe music is largely consistent with their generic usage: it is difficult to think of music that is both strong and not powerful.

Table 7.8: Most correlated duplets of attributes - intrinsic semantic correlation

Attribute1	Attribute2	Φ
Situation Christmas	Genre Special Occasions	0.91
Textcategory Christmas	Genre Special Occasions	0.89
Genre Well-known music	Popularity Popularity high	0.68
Mood strong	Character powerful	0.68
Mood bursting	Character setting off	0.65
Mood harmonious	Character well-balanced	0.60
Situation Sex	Mood erotic/sexy	0.56
Character robotic	Mood technical	0.55
Textcategory Love	Textcategory Relationship	0.55
Situation Fast Ride	Situation Action/Fast Cuts/Hectic	0.53
Situation Middle Ages	Situation historic	0.52
Mood negative	Character mean	0.51
Mood aggressive	Character mean	0.51
Mood mechanical	Character robotic	0.50
Situation Tropical	Country Jamaica	0.50
Situation Computer Animation	Situation SciFi/Futuristic	0.50

Extrinsic music knowledge

The correlations observed in Table 7.9 are probably the most interesting and useful in the context of content-based musical systems. They reveal associations which are not intrinsic properties of the words used to describe music, but which are extrinsic properties of the music domain being described, e.g.

- between musical genres: "Rap" and "Hip hop"

- between genres and countries: “Bossa Nova” and “Brazil”
- between genres and instruments: “Hip Hop” and “Spoken vocals”
- between genres and epoque: “Rag time” and “1930’s”
- between setups and instruments: “Rock band” and “Electric guitar”
- between genres and mood/character: “Jazz” and “Warm”, “Metal” and “Mean”
- between instruments and countries: “Tabla” and “India”
- between instruments and mood: “Transverse flute” and “Warm”

Some of these relations capture historical or cultural knowledge (“rock uses guitars”), but also more subjective aspects linked to perception of timbre (“flute sounds warm”).

Conclusion

The results in Tables 7.5 to 7.9 show that there are important amounts of correlation between high-level descriptors, independently of their relation to timbre. Some of these correlations capture psycholinguistical semantic associations (“a powerful song is a strong song”), but also historical and cultural knowledge (“rock uses guitars”), and more subjective aspects linked to perception of timbre (“flute sounds warm”). This suggests that very many high-level cultural descriptions of music *can* indeed be grounded to timbre similarity, by exploiting such higher-level correlations with timbre-based attributes. We now describe a technique, decision trees, able to do so.

7.2.3 Exploiting correlations with decision trees

A possible way to exploit correlations between attributes is to use decision trees (Quinlan (1993)). A decision tree predicts the value of a given attribute (the *category* attribute) on the basis of answers to questions about the other *non-category* attributes. In the tree, each node corresponds to a non-categorical attribute and each arc to a possible value of that attribute. A leaf of the tree specifies the

expected value of the categorical attribute for the records described by the path from the root to that leaf. Several efficient algorithms (ID3, C4.5) exist to learn precise, compact and robust decision trees from training data. In this work, we use the implementation of C4.5 provided by the Weka library (Witten and Frank (2005)).

Figure 7.3 shows a typical decision tree learned on the attribute database, using the "Variant natural/acoustic" as categorical attribute to predict, and the whole set of other attributes as non-categorical attributes. The tree has been automatically pruned to only use 14 attributes, and gives a prediction precision of 0.71 on positive examples and 0.82 on negative examples (using 10-fold cross validation). Decision trees have been popular in the data mining community notably because of the fact that they produce human-readable decision rules (which a priority order). In the example given in Figure 7.3, we see that a song is likely to be "natural/acoustic"

- if it is not aggressive
- if it is from the 50's (where little amplification was used)
- if there is a singer, but not in the "rock" style
- if it's a folk or a jazz band that performs it
- if not, then if it doesn't use guitar with distortion, etc.

Note that this tree is only able to predict the value of "Variant natural/acoustic" if we have access to all the values of the 14 other non-categorical attributes used here. Therefore, this is of little use as such. However, we will show in the next section that we can use Timbre Similarity Inference to bootstrap the automatic categorization with estimates of a few timbre-grounded attributes, and then use these estimates in decision trees to predict non-timbre correlated attributes.

7.3 An operational model for grounding high-level descriptions

Experiment 9 used a technique to infer the value of attributes which are reasonably correlated with the timbre of the music being described. The technique provides very precise estimates for attributes

```
Variant aggressive = true: false (23.0)
Variant aggressive = false
    Aera/Epoch 1950-1960 = true: true (21.0)
    Aera/Epoch 1950-1960 = false
        Genre Singer/Songwriter = true
            Situation Action/Fast Cuts/Hectic = true
                Style Rock = true: false (2.0)
                Style Rock = false: true (2.0)
            Situation Action/Fast Cuts/Hectic = false: true (67.0/7.0)
        Genre Singer/Songwriter = false
            Musical Setup Folk Band = true: true (25.0/2.0)
            Musical Setup Folk Band = false
                Style Jazz = true: true (25.0/3.0)
                Style Jazz = false
                    Genre Live Classic = true
                        Main Instruments Guitar (distortion) = true: false (4.0)
                        Main Instruments Guitar (distortion) = false: true (29.0/4.0)
                    Genre Live Classic = false
                        Situation Fight = true: false (11.0)
                        Situation Fight = false
                            Style Soul = true: true (9.0/3.0)
                            Style Soul = false
                                Aera/Epoch 1960-1970 = true
                                    Variant live = true: true (2.0)
                                    Variant live = false: false (5.0/1.0)
                                Aera/Epoch 1960-1970 = false
                                    Aera/Epoch 1980-1990 = true: false (4.0)
                                    Aera/Epoch 1980-1990 = false : false:true(10.0/3.0)
```

Figure 7.3: Decision tree for category "Variant natural/acoustic", achieving $p_{true} = 0.71$ and $p_{false} = 0.82$. Figures in parenthesis at each leaf show the number of songs well/misclassified by the corresponding decision rule.

such as homogeneous genre categories or extreme moods like “aggressive” or “warm”. However, it fails on cultural or subjective attributes which bear little correlation with the actual sound of the music being described, such as “TextCategories” or complex moods or characters. **Experiment 10** described the correlations existing between attributes, and proposed a technique (decision trees) able to exploit these correlations to predict values of attributes. This second technique works equally well on timbre or cultural attributes, however it requires the availability of values for non-categorical attributes, to be used as features for prediction. In this section, we propose to build a hybrid automatic categorization system which uses timbre inference as a bootstrap for decision trees. First, we use timbre inference to estimate the values of a few timbre-correlated attributes, and then use decision trees to make further prediction of cultural attributes on the basis of the pool of timbre-correlated attributes.

7.3.1 Algorithm

More precisely, we define the algorithm as an iterative estimation of a set of N attributes $S_A = \{A_k, k \in [0, N-1]\}$. At each iteration i, we produce a set of attribute estimates $\widetilde{S_A}^i = \{\widetilde{A_k}^i, k \in [0, N-1]\}$, where $\widetilde{A_k}^i$ is the estimate of attribute A_k at iteration i. Each attribute estimate is associated with a precision $p(\widetilde{A_k}^i)$. At each iteration i, we define as $best(\widetilde{A_k}^i)$ the best estimate of A_k so far, i.e.

$$best(\widetilde{A_k}^i) = \widetilde{A_k}^m, m = \arg\max_{j \leq i} p(\widetilde{A_k}^j) \tag{7.5}$$

The algorithm is an iterative process:

- $i = 0$: The $\widetilde{A_k}^0$ are built using timbre inference, as described in Experiment 9. Timbre-correlated attributes are typically estimated with good precision $p(\widetilde{A_k}^0)$, while cultural and subjective attributes are poorly estimated.

- $\forall i \geq 1$: $\widetilde{A_k}^i$ is built using a decision tree using A_k a categorical attribute, and a set of non-

categorical attributes $\mathcal{F}_k{}^i$ defined as:

$$\mathcal{F}_k{}^i = \{best(\widetilde{A_l{}^{i-1}})/l \neq k, p(best(\widetilde{A_l{}^{i-1}})) \geq \theta\} \tag{7.6}$$

where $0 \leq \theta \leq 1$ is a precision threshold. $\mathcal{F}_k{}^i$ contains the best estimate so far (up to iteration $i-1$) of every attribute other than A_k, provided that its precision be greater than θ. The algorithm thus constructs successive estimates for each attribute using decision trees on the best estimates at previous steps, the whole process being bootstrapped by timbre inference.

- Stop condition: When there is no more improvement of any attribute estimate, i.e. the set of all $best(\widetilde{A_k{}^i})$ reaches a fixed point.

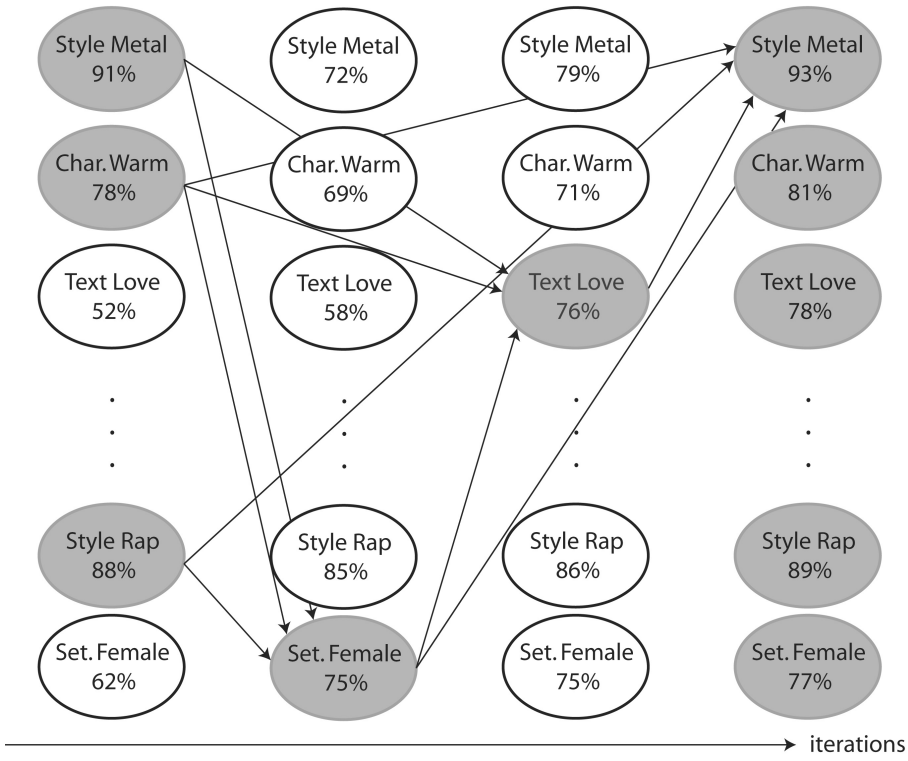

Figure 7.4: An example scenario of iterative attribute estimation

Figure 7.4 illustrates a possible scenario of the above process, using a set of attributes including "Style Metal", "Character Warm", "Style Rap" (which are timbre-correlated attributes as seen in

Experiment 9) and "TextCategory Love" and "Setup Female Singer", which are poor timbre estimated (the former being a cultural description, and the latter being too complex to be precisely described by timbre). The first set of estimates $\widetilde{S_A{}^0}$ is built using timbre inference, and logically contains good estimates of the timbre-correlated attributes, and poor estimates for the others. At iteration 2, we estimate each of the attributes using a decision tree on the timbre estimates (only keeping estimates above $\theta = 0.75$). For instance, we estimate "Style Metal" by using a decision tree on "Character Warm" and "Style Rap", which yields a poorer estimate that the original timbre inference. Similarly, we estimate "Setup Female Singer" using a decision tree on "Style Metal", "Character Warm" and "Style Rap": this yields an estimate which is better than the original timbre inference. At the next iteration, the just produced estimate of "Setup Female Singer" (which happens to be above threshold θ) is used in a decision tree to give a good estimate of "TextCategory Love" (as e.g. the knowledge of whether the singer is a female may give some information about the lyric contents of the song). At the next iteration, all best estimates so far are used in a decision tree to yield an estimate of "Style Metal" which is even better than the original timbre inference (as it uses some additional cultural information).

7.3.2 Preliminary results

Table 7.10 shows the results of the above algorithm on a set of 45 randomly chosen attributes, using $\theta = 0.7$. We observe that for 10 attribute estimates, the precision improves by more than 10% (absolute), and that 15 estimates have a final precision greater than 70%. Cultural attributes such as "Situation Sailing" or "Situation Love" can be estimated with reasonable precision, whereas their initial timbre estimate was poor. It also appears that two "Main Instrument" attributes (guitar and choir), that were surprinsingly bad timbre correlates, have been refined using correlations between cultural attributes. This is in neat accordance with the paradoxical nature of timbre judgements mentionned above (see Section 7.1.3).

The overall precision improvement on the set of attributes depends on the overall correlation of the set of attributes: it is likely that better results could be achieved using e.g. the full set of

attributes, since it would allow the decision trees to exploit stronger correlations that the one found here. It also depends on the quality of the original timbre-correlated estimates, which are used for bootstrap. Here, the randomly chosen set was quite poor on this respect (no initial estimate is greater than 75%).

Table 7.9: Most correlated duplets of attributes - extrinsic musical knowledge

Attribute1	Attribute2	Φ
Character warm	Genre Jazz Standard	0.84
Country Africa	Style Afro	0.80
Style Rap	Style Hip Hop	0.80
Style Post Bop	Style Bossa Nova	0.79
Style Ska Rock	Style Punkrock	0.77
Main Instruments Vocals (Spoken; Rap)	Style Rap	0.75
Main Instruments Vocals (Spoken; Rap)	Style Hip Hop	0.70
Style Nu Jazz	Character collage	0.70
Main Instruments Bandoneon	Main Instruments Musette	0.70
Genre Ethnic Music	Language other	0.65
Style Ragga	Main Instruments Synth Drums	0.62
Style Reggae	Country Jamaica	0.62
Country Brazil	Style Bossa Nova	0.59
Style Jazz (trad.)	Main Instruments saxophone	0.59
Style Latin	Main Instruments Latin Percussion	0.57
Mood ironical	Style Ska Pop	0.57
Musical Setup Rock Band	Main Instruments Guitar (distortion)	0.54
Style Jazz (trad.)	Character warm	0.54
Character mean	Style Metal	0.53
Musical Setup Big Band	Aera/Epoch 1940-1950	0.52
Main Instruments Brass	Musical Setup Wind Ensemble (Winds)	0.51
Style Reggae	Genre Ethnic Music	0.51
Main Instruments transverse flute	Character warm	0.51
Character minimalistic	Style Techno	0.51
Main Instruments Tabla	Country India	0.50
Main Instruments Guitar (distortion)	Character distorted	0.50
Genre Revue	Style Ragtime	0.49
Genre Comedy/Cartoon	Style Ragtime	0.49
Genre Comedy/Cartoon	Aera/Epoch 1930-1940	0.49
Style Swing-Fox	Style Foxtrot	0.49

Table 7.10: Set Optimization of 45 attribute estimates

Attribute	$p(\widetilde{A_k^0})$	$p(\widetilde{A_k^{i_{final}}})$	i_{final}	Improvement
Situation Sailing	0.48	0.71	10	0.23
Situation Flying	0.49	0.64	3	0.15
Situation Rain	0.50	0.64	9	0.14
Main Instrument Guitar	0.60	0.69	4	0.09
Situation Sex	0.59	0.68	11	0.09
Situation Love	0.63	0.70	3	0.07
Textcategory Love	0.61	0.67	11	0.06
Situation Party/Dance	0.60	0.66	6	0.06
Tempo medium - Andante	0.59	0.64	4	0.05
Character slick	0.65	0.69	11	0.04
Aera/Epoch 90s	0.71	0.75	13	0.04
Character well-balanced	0.62	0.66	6	0.04
Rhythmics rhythmic	0.64	0.68	4	0.04
Genre Dancemusic	0.65	0.68	12	0.03
Mood dreamy	0.64	0.67	2	0.03
Style Pop	0.71	0.74	6	0.03
Mood positive	0.58	0.61	6	0.03
Mood harmonious	0.62	0.65	4	0.03
Main Instruments Vocals (Choir)	0.60	0.63	13	0.03
Dynamics dynamic (up+down)	0.61	0.63	5	0.02
Textcategory Associations	0.57	0.59	10	0.02
Variant expressive	0.62	0.64	2	0.02
Musical Setup Pop Band	0.72	0.74	7	0.02
Textcategory Poetry	0.57	0.59	10	0.02
Character friendly	0.65	0.67	6	0.02
Character repeating	0.63	0.64	9	0.01
Rhythmics groovy	0.63	0.64	4	0.01
Mood romantic	0.69	0.70	9	0.01
Textcategory Wisdom	0.58	0.59	4	0.01
Textcategory Romantics	0.65	0.66	14	0.01
Aera/Epoch 1990-2000	0.76	0.76	1	0
Affiliate International	0.72	0.72	1	0
Character creamy	0.72	0.72	1	0
Genre Ballad	0.72	0.72	1	0
Main Instruments Vocals	0.71	0.71	1	0
Main Instruments male	0.71	0.71	1	0
Genre Mainstream	0.71	0.71	1	0
Genre Adult Contemporary (AC)	0.71	0.71	1	0
Mood erotic/sexy	0.70	0.70	1	0
Main Instruments Synthesizer	0.69	0.69	1	0
Language English	0.69	0.69	1	0
Main Instruments SFX (Sound Effects)	0.67	0.67	1	0
Character melodical	0.62	0.62	1	0
Textcategory Conciousness	0.6	0.6	1	0
Variant staccato	0.57	0.57	1	0

Chapter 8

Conclusion: Toward Cognitive Models

The majority of systems extracting high-level music descriptions from audio signals rely on a common, implicit model of the global sound or *polyphonic timbre* of a musical signal. This model represents timbre as the long-term distribution of the local spectral features, a prototypical implementation of which being Gaussian Mixture Models of Mel-Frequency Cepstrum Coefficients.

This thesis questions the validity of this model. To do so, we have tried to construct an explicit measure of the timbre similarity between polyphonic music textures, by mobilizing all the tools and design heuristics typically at use in Music Information Retrieval research.

With **Experiment 1**, we showed that the precision of measures based on this approach could be optimized to satisfactory levels. However, we described many variants that surprisingly did not lead to any substantial improvement of the measure's precision. Moreover, our simulations suggest the existence of a *glass ceiling* at precision about 70%. The remaining error rate is not incidental, and is indicative of a structural limitation which probably cannot be overcome by such variations on the same theme.

More precisely, modelling the long-term statistical distribution (accounting for time or not - HMMs or GMMs) of the individual "atoms" or "grains" of sound (frames of spectral envelopes), and comparing their global shape constitutes a strong assumption on the underlying cognitive process. While it is clear that the perception of timbre results from an integration of some sort (indi-

vidual frames cannot be labeled independently, and may "come" from very different textures), other important aspects of timbre perception are not covered by this approach.

One surprising finding of **Experiment 1** is that algorithms that account for the time dynamics of the features, e.g. with dynamic programming or hidden Markov models, are at best equivalent to simpler static models. This is at odds with experimental data on the perception of individual instrument notes. **Experiment 2** established that the polyphonic nature of the data is the main reason that ruins computational attempts at modelling feature dynamics. This suggests that the horizontal coding of frames of data, without any account of source separation and selective attention, is a very inefficient representation of polyphonic musical data, and not cognitively plausible. On that respect, more brain-plausible processings such as sparse representations (Georgiev et al., 2005; Daudet, 2006) may provide a fruitful direction for further research.

The most important and novel finding of **Experiment 1** is that the class of algorithms studied in this work tends to create false positives which are mostly always the same songs regardless of the query. In other words, there exist songs, which we call *hubs*, which are irrelevantly close to all other songs.

We established that:

- hubs are distributed according to a scale-free distribution.
- hubs are not a consequence of poor feature representation of each individual frame, but rather an effect of the modelling of the agglomeration of the many frames of a sound texture (**Experiment 3**).
- hubs are not a property of a given modelling strategy (i.e. static vs dynamic, parametric vs non-parametric, etc.) but rather tend to occur with any type of model (**Experiment 4**).
- hubs are not an intrinsic property of certain songs, but that different algorithms distribute the hubs differently on the whole database (**Experiment 5**).
- the hubness of a given song is not an emerging global property of the distribution of its

frames, but rather can be localised to certain parts of the distribution, notably outlier frames in statistical minority (**Experiment 7**).

- hubs are not a property of the class of algorithms studied here which appears regardless of the data being modelled, but only for data with a given amount of heterogeneity, e.g. for polyphonic music, but not for ecological sound ambiances (**Experiment 8**).

This phenomenon of hubs is reminiscent of other isolated reports in different domains, such as Speaker Recognition or Fingerprint Identification, which intriguingly also typically rely on the same features and pattern-recognition algorithms. This suggests that this could an important phenomenon which generalizes over the specific problem of timbre similarity, and indicates a general structural property of the class of algorithms examined here. This of course would require further investigation, for which this study provides a methodological basis, notably by introducing metrics to quantify hubness.

The phenomenon of hubs, and notably the evidence of its important sensibility to certain critical frames (**Experiment 7**), also illustrates one deep discrepancy between all computation models of timbre and human perception. Namely, that all frames are not of equal importance, and that these weights does not merely result of their long-term frequencies(i.e. the corresponding component's prior probability π_m). The "GMM of MFCC" approach essentially builds an *extensional* description of the object being modelled. While this seems to be a sufficient (and efficient) model for the perception of environmental audio textures, music categorization appears to create essentially *intensional* constructs, which are poorly modelled by the models studied here. In particular, some timbres (i.e. here sets of frames) are more *salient* than others : for instance, the first thing than one may notice while listening to a Phil Collins song is his voice, independently of the instrumental background (guitar, synthesizer, etc...). This saliency may depend on the context or the knowledge of the listener and is obviously involved in the assessment of similarity.

Experiments 9 and 10 give further support in the fact that "timbre" judgements are not low-level immediate perceptions, but rather high-level cognitive reasoning which accounts for the evi-

dence found in the signal, but also depends on cultural expectations, a priori knowledge, and context. **Experiment 9** shows that surprisingly few human-made high-level music descriptions, and notably judgements of instrument classes, are directly correlated to low-level timbre similarity. Polyphonic textures as found in popular music are cultural objects whose perception creates expectations based on music that a particular listener already knows. In **experiment 10**, we showed that human judgements could be approximated only by accounting for high-level correlations within a large set of possible categories. Some of these correlations capture psycholinguistical semantic associations ("a powerful song is a strong song"), but also historical and cultural knowledge ("rock uses guitars"), and more subjective aspects linked to perception of timbre ("flute sounds warm"). Much of the music we hear as being "piano music" is really music that *we expect to be* piano music.

These experiments open the way for more careful investigations of the perceptive paradoxes proper to polyphonic music timbre, in which listeners "hear" things that are not statistically significant in the actual signal, and that the low-level models of timbre similarity studied in this work are intrinsically incapable of capturing.

Part III

Synthèse en français - Digest in French

Résumé détaillé

Chapitre 1: Introduction

La grande majorité des systèmes d'extraction de metadonnées haut-niveau à partir de signaux musicaux repose sur un modèle implicite de leur "son" ou *timbre polyphonique*. Ce modèle représente le timbre comme la distribution statistique globale de propriétés spectrales instantanées (*features*), calculées sur des trames de quelques dizaines de millisecondes. Une implémentation prototypique de cette approche utilise des modèles de mélange de gaussiennes et des coefficients cepstraux.

Cette thèse remet en cause la validité de ce modèle.

Pour ce faire, nous construisons ici une mesure explicite de la similitude timbrale entre deux textures polyphoniques. Nous utilisons tout l'attirail méthodologique (outils, algorithmes, heuristiques) developpé dans le domaine de la reconnaissance de formes appliqué à la musique (*Music Information Retrieval*). Nous étudions les propriétés de cette mesure dans une série de dix expériences.

L'**expérience 1** montre que la précision des mesures construites sur le paradigme que nous étudions ici est bornée par un *plafond de verre* empirique, à environ 70% de R-precision. Le taux d'erreur résiduel n'est pas accidentel, mais révèle plutôt des limitations fondamentales, qui ne sauraient être résolues par de futures variations sur le même theme.

Une des conclusions de cette expérience est l'apparente inutilité des variantes algorithmiques

destinées à mieux modéliser la dynamique des attributs instantanés, tels que l'emploi de coefficients dérivés, ou de modèles de Markov cachés. Ceci contredit des nombreuses expériences psychoacoustiques, qui ont démontré l'importance de la dynamique pour la perception humaine des timbres instrumentaux. L'**expérience 2** montre que cette difficulté à modéliser la dynamique des séquences d'attributs instantanés est due au caractère polyphonique des données, et non à leur structure temporelle (en notes, phrases, refrains, etc.). Ce résultat suggére que le codage horizontal des trames audio, qui ne prend pas en compte les différentes sources et leur synchronisation verticale, est une représentation peu efficace pour la musique polyphonique, et peu plausible cognitivement.

Une de nos observations les plus importantes est que la classe d'algorithmes étudiée ici crée de faux-positifs qui sont presque toujours les mêmes chansons, indépendamment de la requête de départ. Entre d'autres termes, il existe pour ces algorithmes des chansons anormalement proches de toutes les autres chansons: de faux centres de gravité (*hubs*). Par une série d'expériences, nous établissons ici que

- ces hubs sont distribués selon une loi de puissance, dite à invariance d'échelle (*scale-free*).
- les hubs ne sont pas la conséquence d'un mauvais comportement des attributs instantanés pris individuellement, mais de la modélisation de leur agglomération à long-terme. Entre d'autres termes, le problème ne vient pas des *features*, mais des modèles (**Expérience 3**).
- les hubs ne sont pas la conséquence d'une stratégie d'apprentissage statistique donnée, mais ont tendance à apparaître pour tous les algorithmes, paramétriques ou non, dynamiques ou non (**Expérience 4**).
- le fait d'être un hub n'est pas une propriété intrinsèque d'une chanson donnée. Différents algorithmes distribuent les hubs de façon différente sur le même corpus de chansons (**Expérience 5**).
- Les hubs ne sont pas la conséquences de modèles peu discriminants. En particulier, les chansons dont la modélisation statistique révèle une grande variance ne sont pas particulièrement

sujettes à devenir des hubs. D'autre part, les modèles dynamiques, plus contraints que les modèles statiques, créent tout autant de hubs que leurs homologues (**Expérience 6**).

- l'importance d'un hub donné n'est pas une propriété émergeant globalement de la distribution de ses attributs instantanés, mais peut être attribuée à certaines parties de cette distribution, notamment les valeurs d'attributs en minorité statistique (**Expérience 7**).

- les hubs ne sont pas une propriété intrinsèque des algorithmes étudiés ici, qui apparaîtrait indépendamment des données modélisées. En particulier, les hubs apparaissent dans des corpus de musique polyphonique, mais pas de sons environnementaux. Nous montrons que la différence entre les signaux "à hubs" et les autres se traduit entre autres par l'hétérogénéité statistique de leur distribution (**Expérience 8**).

Ce phénomène de hubs rappelle curieusement des observations déjà faites dans d'autres domaines que la musique, notamment en identification biométrique de locuteurs ou d'empreintes digitales. Le rapprochement est d'autant plus intéressant à faire que ces technologies emploient typiquement les même techniques algorithmiques que celles que nous étudions ici. Cela incite à penser que ce phénomène est une caractéristique importante de ce type d'algorithmes, et que nos conclusions s'étendent donc au delà du problème spécifique de la similitude timbrale de signaux musicaux. Ce phénomène traduit une situation générale dans laquelle les observations instantanées n'ont pas toutes la même importance perceptuelle, qui en particulier ne dépend pas de leur salliance statistique par rapport à leur distribution à long-terme.

Ce point est précisé dans une dernière partie, où nous donnons des éléments quantitatifs nouveaux pour apprécier le fait que les jugements humains en matière de timbre polyphoniques ne sont pas le résultat immédiat de perceptions bas-niveau, mais plutôt de raisonnement cognitifs évolués, dépendant par exemple du contexte d'écoute et de la culture de l'auditeur. L'**expérience 9** montre que, de façon surprenante, très peu de métadonnées musicales haut-niveau, et en particulier les descriptions de l'instrumentation, sont corrélées directement au timbre tel que nous le modélisons ici. Dans l'**expérience 10**, nous montrons que ces processus de description ne peuvent être ap-

proximés algorithmiquement qu'en prenant en compte des corrélations avec des attributs plus culturels et subjectifs, comme les classes d'émotions. Ces corrélations traduisent des associations sémantiques de l'ordre de la psycholinguistique ("une chanson *puissante* (powerful) est une chanson *forte* (strong)"), mais aussi des connaissances historiques ou culturelles ("le *rock* utilise de la guitare électrique"), et même des aspects plus subjectifs liés à la perception du timbre ("la flute crée un son *chaleureux* (warm)").

En d'autres termes, nous "entendons" quotidiennement dans la musique polyphonique des choses qui ne sont pourtant pas présentes de façon significative (statistiquement) dans le signal sonore. La musique que nous *entendons* être du piano est surtout de la musique que nous nous *attendons* à être du piano. Ces paradoxes statistico-perceptifs, dont l'existence devrait maintenant être établie de façon rigoureuse, expliquent en grande partie le désaccord entre les modèles algorithmiques étudiés ici et la perception humaine.

PARTIE I: EPISTEMOLOGIE

Chapitre 2: Les Dimensions du Timbre

2.1 Tout sauf la hauteur et l'intensité

Le timbre est défini par l'American Standards Association comme cet attribut de la perception qui permet à l'auditeur de distinguer deux sons ayant la même hauteur (*pitch*) et la même intensité (*loudness*). Le timbre est un attribut continu (e.g. un son peut être *plus ou moins* "brillant"), et multidimensionnel (on ne saurait trouver un même échelle où ranger tous les sons allant, par ex., du piano à la trompette).

2.2 Etudes psychophysiques

De nombreuses études psychoacoustiques ont tenté d'établir une cartographie de l'espace des timbres musicaux, et d'en identifier les principales dimensions. La plupart utilise des techniques de positionnement multidimensionnel (*multidimensional scaling*) de données de similitude entre extraits musicaux, collectées sur des sujets humains. Elles concluent majoritairement à l'existence de trois dimensions principales, corrélées respectivement au centre de gravité spectral (*spectral centroid*), au logarithme du temps d'attaque (*log-attack time*), et aux variations temporelles de l'envelope spectrale (*spectral irregularity*, *spectral flux*).

2.3 Reconnaissance automatique de sons instrumentaux

Ces études ont motivé la conception d'un grand nombre de systèmes identifiant automatiquement l'instrument de l'enregistrement d'une note individuelle, ou d'une phrase monoinstrumentale. La plupart de ces systèmes utilisent une combinaison d'attributs temporels (typiquement temps d'attaque et coefficients de corrélation) et spectraux (typiquement coefficients cepstraux - voir plus loin). La stratégie la plus simple pour comparer ces attributs entre signaux sonores est d'utiliser la distance euclidienne, et un algorithme de plus proches voisins. Toutefois, de nombreux systèmes

de classification utilisent maintenant des modèles plus complexes, comme les mélanges de gaussiennes, ou les machines à vecteurs-support. Enfin, de récentes contributions ont montré l'intérêt pour de tels sons instrumentaux d'employer des modèles dynamiques, tels que les modèles de Markov cachés, ou des mesures d'entropies sur des séquences d'attributs préalablement quantisés.

2.4 Vers les textures polyphoniques

L'immense majorité des études précédentes se sont intéressées à des échantillons sonores correspondant à des notes individuelles, jouées par un seul instrument. Toutefois, le contexte émergent de la distribution électronique de musique, *à la iTunes*, et de l'explosion des capacités de stockage sur les ordinateurs personnels, crée une forte demande de modèles de signaux applicables à des textures de plusieurs secondes (voire des morceaux entiers), et fortement polyphoniques. En particulier, la notion de "timbre polyphonique", ou de musique qui "sonne comme" e.g. du *Boulez*, *Madonna*, ou *Chet Baker*, semble être une abstraction bien adaptée à ces nouveaux besoins. Le "son des musiques" est en particulier une façon naturelle de représenter la préférence d'un utilisateur pour telle ou telle époque, configuration, style, genre ou esprit musical ("le *Chick Corea* des années 70","les musiques de films à la *Out of Africa*", "tout ce qui sort sur le label *Ninja Tunes*", etc.).

Toutefois, les conclusions des études psychoacoustiques, relayées par les systèmes de reconnaissance mono-instrumentaux, ne s'étendent pas facilement aux textures polyphoniques, pour un certain nombre de raisons:

- les descripteurs monophoniques, de type centre de gravité spectral, ne sont pas évaluables de façon additive sur des textures polyphoniques, i.e. la résultant polyphonique n'est pas la somme pondérée des descripteurs des sources individuelles

- l'asynchronicité des sources superposées rend compliqué le calcul de descripteurs temporels, comme le temps d'attaque.

- les séquences de plusieurs notes ont une influence contextuelle sur la perception du timbre de

chaque note, et réciproquement, le timbre a une influence sur la perception des séquences.

- une grande partie du “son” des musiques actuelles tient plus du bruit coloré (guitare saturée, caisse claire, effets de production), que du signal harmonique traditionnellement étudié dans le contexte mono-instrumental.

2.5 Des modèles implicites

Dans ce contexte, la plupart des systèmes d’extraction de métadonnées à partir de signaux polyphoniques ont adoptés la même approche pragmatique et générique, fondée sur la modélisation de la distribution statistique globale de propriétés spectrales instantanées. Cette approche se décline en une infinité de variantes, portant soit sur les attributs utilisés (par exemple les coefficients cepstraux, ou les attributs MPEG7), soit sur les modèles (paramétriques ou non, dynamiques ou non). Ce type d’algorithmes constitue plus du quart des contributions à la conférence ISMIR (International Conference on Music Information Retrieval) depuis sa création en 2000. Toutefois, toutes ces variantes se fondent sur la même hypothèse, rarement explicitée, que la perception du timbre d’une texture polyphonique correspond à la salience statistique de ses attributs instantanés les plus représentés.

La validité de cette hypothèse est difficile à évaluer d’aprés le corpus de travaux existants, notamment car

- Chaque contribution n’évalue pas la précision d’un modèle en terme de similarité timbrale, mais de descriptions plus haut-niveau, comme le genre, dont le lien avec le timbre n’est pas toujours facile à établir. Par exemple, la difficulté qu’a un modèle donné à discriminer des signaux de genre “classique” et “jazz” peut révéler les limitations du modèle timbral sous-jacent, mais tout autant l’incohérence de la taxonomie utilisée: les arrangements orchestraux des jazzmen *Gil Evans* ou *Carla Bley* sont timbralement plus proches de compositions d’*Alan Berg* ou de *Debussy* que de *Charlie Parker*.

- De façon symétrique, le succés d’un modèle donné sur une tâche de classification peut être dûe à un mécanisme d’apprentissage supervisé efficace, comme un SVM ou un mélange d’experts,

et ce *malgré* une représentation sous-jacente de la simitude timbrale qui peut être limitée.

- Finalement, très peu d'études avant la notre ont comparé différents algorithmes sur une même base de données, ce qui évidemment donne peu de crédit aux affirmations individuelles de supériorité de tel ou tel algorithme.

Cette thèse propose donc d'étudier frontalement la validité de l'approche commune à tous ces travaux, en construisant une mesure explicite de la similitude timbrale entre deux textures polyphoniques, sans entacher nos conclusions de corrélations non maîtrisées entre des concepts plus haut-niveau comme le genre. Nous étudions les propriétés de cette mesure dans une série de dix expériences, dont les conclusions ont des implications pour la modélisation du timbre polyphonique, mais aussi pour les systèmes d'extraction plus haut-niveau qui utilise implicitement le même modèle. Une partie de nos résultats, l'existence de faux centre de gravités ou "hubs", semble même se généraliser au domaine plus général de la reconnaissance de formes.

Chapitre 3: Les Dimensions des *Modèles* timbraux

La majorité des systèmes d'extraction de métadonnées à partir de signaux musicaux polyphoniques sont fondés sur le même modèle implicite de timbre, i.e. la distribution statistique globale de propriétés spectrales instantanées. Cette approche se décline dans la littérature en une infinité de variantes. Nous proposons de décrire cet espace de variations à partir d'un algorithme prototypique ("le GMM de MFCC"), auquel on applique un certain nombre de transformations, à la manière des modèles de conception réutilisables en ingénierie logicielle (*design patterns*).

3.1 L'algorithme prototypique

Comme vu précédemment, il s'agit d'une mesure algorithmique de la similitude timbrale globale entre deux signaux musicaux. La mesure suit le même paradigme que les nombreux systèmes d'extraction de métadonnées haut-niveau, en ce qu'elle est basée sur la modélisation statistique des distributions globales d'attributs spectraux. Toutefois, elle modélise explicitement la notion de similitude perceptive, plutôt qu'une catégorisation plus haut-niveau dont l'implicite corrélation au timbre ne serait pas maitrisée.

Le signal est découpé en trames. Sur chaque trame, nous estimons l'enveloppe spectrale du signal en calculant un ensemble de coefficients mel-cepstraux. Le cepstre (une anagramme de "spectre") est la transformée de Fourier inverse du logarithme du spectre. On appelle mel-cepstre le cepstre calculé aprés un rééchelonnement du spectre de Fourier sur une échelle de fréquence perceptive, non-linéaire: l'échelle Mel. Les coefficients issus de la transformée inverse du spectre sont appelés coefficients mel-cepstraux (*Mel-Frequency Cepstrum Coefficients* - MFCCs). On en garde un nombre N. Les coefficients cepstraux produisent une représentation compacte du spectre lissé, d'autant plus précise que N est grand.

La distribution des coefficients mel-cepstraux (dans un espace de dimension N) est ensuite modélisée par un modèle de mélange de gaussiennes (*Gaussian Mixture Model* - GMM). Un GMM est un estimateur paramétrique de distribution de probabilité, sous la forme d'une somme pondérée

de *M* densités gaussiennes appelées composantes du mélange. Les paramètres du GMM (poids, moyenne et covariance de chacune des *M* composantes) sont estimés par un algorithme itératif d'optimisation de la log-vraisemblance, E-M (pour *Expectation - Maximisation*).

Chaque chanson est donc modélisée par un GMM, qui peut être comparé avec les modèles d'autres chansons. Nous utilisons pour celà un approximation par Monte-Carlo de la divergence de Kullback-Leibler, une distance naturelle entre densités de probabilité qui n'a pas de forme analytique pour les mélanges de gaussiennes. L'estimateur consiste à générer un grand nombre de points *n* (en dimension *N*) à partir de l'un des modèles, et de moyenner le logarithme de la probabilité de ces points selon l'autre modèle (puis de faire l'opération inverse pour symétriser la mesure).

3.2 Transformations et Heuristiques employées en MIR

Une étude bibliographique montre qu'à cet algorithme prototypique, ne sont typiquement appliquées qu'un nombre fini de transformations et d'heuristiques de recherche, dont nous proposons ici un catalogue. Il est intéressant de constater que ces mêmes heuristiques ont souvent été utilisées dans le domaine plus ancien de la reconnaissance automatique de la parole.

Optimiser les paramètres des attributs: Typiquement, cela revient à modifier la dimension *N* de l'espace dans lequel se fait la modélisation. Une plus grande dimension permet souvent de distinguer des détails plus précis, ou de mieux décrire le signal de départ. Toutefois, une trop grande dimension rend la modélisation statistique difficile, car elle exige un nombre exponentiel d'exemples d'apprentissage. Ce phénomène est connu sous le nom de "malédiction des grandes dimensions" (*curse of dimensionality*). Le choix des paramètres optimaux résulte donc d'un compromis.

Optimiser les paramètres du modèle: De manière similaire, les paramètres d'un modèle statistique (par ex. le nombre *M* de composantes gaussiennes à apprendre dans un GMM) influent sur le pouvoir d'expression du modèle. Un modèle de plus grande dimension sera capable de représenter une distribution non triviale de façon plus fidèle. Toutefois, un modèle

"plus complexe" que la distribution à modéliser (ce qui peut recevoir une définition formelle, dans le cadre par ex. de la dimension de Vapnik) risque de sur-représenter les exemples d'apprentissage et de très mal se généraliser à des exemples non encore observés. Ce problème est connu sous le nom de sur-apprentissage (*overfitting*). Là encore, donc, le choix des paramètres optimaux d'un modèle résulte d'un compromis.

Remplacer des attributs équivalents: Les attributs utilisés dans la littérature sont souvent regroupés en ensembles équivalents, selon le type d'information qu'ils encodent à partir du signal. Une première taxonomie distingue les attributs temporels (issus par exemple du profil d'énergie, comme le temps d'attaque) des attributs spectraux (calculés à partir du spectre de Fourier). Parmi les attributs spectraux, on peut distinguer les encodages fondés sur la décomposition du spectre de Fourier en moments statistiques (centrode - *centroid*, variance - *spread*, asymétrie - *skewness*, aplatissement - *kurtosis*, etc.) et la décomposition par Fourier à nouveau, qui produit les coefficients ceptraux.

Remplacer des modèles équivalents: De façon similaire, des classes d'équivalence de modèles statistiques tendent à émerger à l'usage en fonction de leurs hypothèses structurelles sous-jacentes. Une distinction possible considère les modèles statiques d'un côté, qui ne prennent pas en compte l'ordonnancement temporel des données d'apprentissage (par ex. les histogrammes, GMM, k-plus-proches-voisins), et les modèles dynamiques de l'autre, qui modélise à la fois la distribution statique et son évolution dans le temps (par ex. modèles de Markov cachés ou réseaux de neurones récurrents). Une autre distinction importante considère les algorithmes paramétriques d'une part, qui modélise une densité de probabilité comme une fonction dont il faut optimiser les paramètres (par ex. les moyennes et covariances d'un GMM), aux modèles d'estimation non-paramétriques qui applique la même représentation à tout type de données (par ex. un histogramme).

Dériver des attributs par composition fonctionnelle: Il est courant de modifier un attribut standard en lui appliquant un pré-traitement (tel qu'un filtrage préliminaire du signal), post-

traitement (en prendre la dérivée) ou en modifiant la chaîne de traitement intermédiaire. Par ex. l'attribut de contraste spectral (*Spectral Contrast*) modifie l'algorithme standard d'extraction des MFCCs en remplaçant la moyenne du spectre faite sur chaque bande de fréquence Mel par sa variance.

Emprunter à d'autres domaines: Il est courant d'emprunter une technique, typiquement un attribut, à un autre domaine d'application de la reconnaissance de formes, lorsqu'elle s'est avérée efficace. Les MFCCs, omniprésents aujourd'hui pour la musique, sont à l'origine un emprunt fait à la reconnaissance de parole, qui elle même les avait adaptés de techniques utilisées pour l'analyse de signaux sismiques.

Modéliser la dynamique: De nombreuses publications étendent le modèle de base afin de prendre en compte la dynamique temporelle des attributs. La modification peut se faire au niveau de l'extraction d'attributs, par exemple en substituant les attributs par leur dérivée temporelle (*delta-coefficients*), leurs statistiques sur des fenêtres temporelles de plusieurs trames (*texture windows*), ou encore leur transformée de Fourier à l'échelle de plusieurs secondes. La modification peut aussi porter sur la modélisation statistique des attributs, en utilisant des modèles dynamiques comme les modèles de Markov cachés.

Intégrer un raisonnement haut-niveau: Une autre direction de recherche est d'intégrer de tels algorithmes basés sur le signal dans un système plus large, qui rajoute une couche de raisonnement plus haut niveau, par exemple sur la structure temporelle à long-terme. L'exemple typique de cette stratégie est l'utilisation en reconnaissance de la parole de modèles de grammaire permettant de désambiguïser les décisions faites localement à partir des seules séquences d'attributs. L'équivalent en musique est de représenter la succession temporelle des notes en transitoires et états stationnaires, ou les changements de tonalités, etc.

3.3 Conclusion

Ce catalogue d'heuristiques de recherche a une utilité rhétorique, car nous l'utilisons dans le chapitre suivant pour structure l'exploration de l'espace des modèles timbraux, afin d'en estimer la précision. Toutefois, cette présentation a aussi valeur pédagogique, en ce qu'elle aide à appréhender l'importante littérature du domaine, qui se caractérise par un cheminement très incrémental dû à une certaine maturité. Enfin, ces heuristiques posent la question d'une formalisation du savoir-faire spécifique au domaine et de la possible automatisation d'un processus d'aide à la recherche, à la manière du système EDS développé à SONY CSL.

PARTIE II: EXPERIENCES

Chapitre 4: Le plafond de verre (Expérience 1)

Dans ce chapitre, nous testons la validité de l'hypothèse sous-jacente au modèle de timbre implicitement utilisé par la majorité des systèmes d'extraction de métadonnées. Pour cela, nous explorons l'espace des algorithmes décrits précédemment, afin d'en optimiser la précision.

4.1 Expérience

Nous implémentons et testons ici plus de 500 variantes algorithmiques, découlant des heuristiques décrites plus haut, et que nous résumons ici:

- Paramètres de l'algorithme original: taux d'échantillonnage, taille des trames, nombre *N* de MFCCs extraits de chaque trame, nombre *M* de composantes gaussiennes, nombre *n* de tirages pour l'approximation de Monte-Carlo
- Descripteurs spectraux MPEG7: remplacement total ou partiel des MFCCs par les descripteurs *SpectralCentroid*, *SpectralSpread*, *SpectralKurtosis*, *SpectralSkewness*, *SpectralFlatness*, *SpectralRolloff*, et *SpectralFlux*.
- Autre mesure de distance: remplacement de Monte-Carlo par plusieurs variantes de la distance *Earth Mover*.
- Modèles non-paramétriques: remplacement des GMMs par plusieurs variantes d'histogrammes, notamment aprés quantisation vectorielle des MFCCs.
- Pré et post-traitement inspirés de la reconnaissance de la parole: opérations de *ZMeanSource*, Pré-emphase, *Dither*, *Liftering*, *Cepstral Mean Compensation* et ajout du coefficient d'ordre 0.
- Variantes de MFCCs: dont plusieurs implémentations de contraste spectral (*Spectral Contrast*).

- Emprunt à l'analyse de textures picturales: notamment l'emploi de matrices de co-occurrences
- Coefficients dérivés et dérivés-secondes
- *Texture Windows*: cad la substitution des MFCCs par leurs statistiques sur des fenêtres de plusieurs secondes.
- Modèles de Markov cachés: et optimisation de leurs paramètres (nombre d'états, nombre de composantes gaussiennes par état).
- Modèles haut-niveau: prenant en compte la notion de note formée d'un transitoire et d'un état stationnaire.

4.2 Méthode

Nous utilisons une base de données de 350 titres construite pour représenter explicitement la notion de similitude timbrale. La base comprend 37 groupes de chansons du même artiste, optimalement similaires entre elles, et optimalement distant les uns des autres. La précision d'un algorithme est définie comme le rapport moyen du nombre de chansons du même groupe que la chanson-requête S obtenues dans les k plus proches voisins de S, quand k est égal à la cardinalité du groupe de S. Cette mesure est appelée R-précision dans la communauté d'*Information Retrieval*.

4.3 Outils

Au total, plus de 500 variantes algorithmiques ont été testées, nécessitant pour chacune le stockage et l'analyse d'une matrice de similitude complète entre les chansons de la base de données. Cette étude, d'amplitude inégalée à notre connaissance, n'est rendu possible que par:

- la conception d'une architecture logicielle dédiée (l'API MCM), et d'une plateforme d'expérimentation se doublant d'un outil de navigation dans des bases de données musicales (le Music Browser), réalisations collectives du groupe musique de Sony CSL

- des efforts d'implémentation des différents algorithmes, pour passer de prototypes Matlab à des versions optimisées, propriétaires, en C, surpassant en vitesse les différents codes existants en accés libre.

- un nouvel algorithme de recherche de plus proches voisins, permettant des gains en vitesse supérieurs à 3000%

4.4 Résultats

L'exploration de l'espace des modèles décrits ci-dessus permet d'améliorer la *R*-précision de plus de 15% par rapport aux réglages initiaux, pour un maximum de 65.2 % *R*-précision. La mesure optimale compare des GMMs de $M = 50$ composantes, calculés pour $N = 20$ MFCCs, additionnés du coefficient d'ordre 0. Les modèles sont comparés par Monte-Carlo, avec $n = 2000$ tirages aléatoires.

Ces 65% de précision peuvent sembler un piètre résultat. Il faut toutefois rappeler que le critère d'évaluation sous-estime la qualité de la mesure en considérant comme incorrects les documents d'un autre groupe que la chanson-requête, qui peuvent toutefois être corrects perceptivement. A titre d'illustration, la mesure optimale trouvée ici a été implémentée par Elias Pampalk et a remporté le concours de classification automatique d'artistes lors de la conférence ISMIR 2004.

Toutefois, le taux d'erreur résiduel n'est pas accidentel, mais révèle plutot des limitations fondamentales:

- La modélisation dynamique, que ce soit par coefficient dérivés, *texture windows*, modèles de Markov cachés ou matrices de co-occurrence, n'améliore pas la précision. Ce comportement est paradoxal, car il est connu depuis les premières expériences psychoacoustiques sur la perception humaine des timbres instrumentaux que les descripteurs dynamiques comme le temps d'attaque jouent un rôle fondamental. Le chapitre suivant décrit une expérience permettant de mieux comprendre cette observation.

- L'expérimentation montre qu'à part un petit nombre de paramètres cruciaux (comme le taux

d'échantillonnage), le choix des paramètres et l'utilisation de telle ou telle variante n'a que peu d'influence sur la précision de la mesure. Les traitements sophistiqués, notamment, sont au mieux équivalents à leur homologues plus basiques.

- Ces résultats suggérent aussi que la précision découlant de telles variations sur le même thème (distribution statistique globale de propriétés spectrales instantanées) est bornée par une asymptote empiriquement estimée à 70% de *R*-précision.

- Une de nos observations les plus importantes est que la classe d'algorithmes étudiée ici crée de faux-positifs qui sont presque toujours les mêmes chansons, indépendamment de la requête de départ. Entre d'autres termes, il existe pour ces algorithmes des chansons anormalement proches de toutes les autres chansons: de faux centres de gravité (*hubs*). Ce phénomène semble être une caractéristique importante de ce type d'algorithmes, qui s'étend au delà du problème spécifique de la similitude timbrale entre signaux musicaux. Nous consacrons par la suite un chapitre à son étude plus approfondie.

Chapitre 5: De l'utilité de la dynamique (Expérience 2)

5.1 Le paradoxe de la modélisation dynamique

Une des conclusions les plus surprenantes de notre étude est l'apparente inutilité des variantes algorithmiques destinées à mieux modéliser la dynamique des attributs instantanés, tels que l'emploi de coefficients dérivés, ou de modèles de Markov cachés. Ceci contredit des nombreuses données expérimentales en psychoacoustique, qui ont démontré l'importance de la dynamique pour la perception humaine des timbres instrumentaux.

5.2 Hypothèses

Trois causes principales peuvent expliquer les difficultés computationnelles à modéliser la dynamique des attributs instantanés dans le cas de textures polyphoniques:

H1 Soit la dynamique des attributs est impossible à représenter algorithmiquement à l'intérieur même d'une note, par ex. à cause d'une trop grande variabilité entre notes. Ceci est improbable, car la possibilité d'une telle modélisation a été établie par plusieurs algorithmes d'identification automatique d'instrument.

H2 Soit dés lors, c'est la dynamique des trames de signaux *polyphoniques* qui est difficile à modéliser, par ex. à cause du masquage spectral entre plusieurs sources et de leur désynchronisation.

H3 Soit enfin, c'est la dynamique de la *succession* des notes et événements musicaux dans une texture qui est difficile à modéliser, à cause par ex. de la structure à long-terme d'une morceau de musique, comme ses changements d'instrumentation.

5.3 Méthodes

Nous proposons de discriminer ces trois hypothèses en comparant les performances d'algorithmes statiques et dynamiques sur deux types de signaux:

- des enregistrements monophoniques, mono-instrumentaux de notes individuelles (DB1)
- des enregistrement polyphoniques de notes individuelles segmentées automatiquement à partir de textures polyphoniques (DB2)

L'hypothèse H1 serait confirmée si l'on établit que les algorithmes dynamiques sont moins efficaces que les algorithmes statiques sur DB1 (ce qui serait, au passage, en contradiction avec les précédents résultats dans la littérature). H2 serait quant à elle confirmée si la modélisation dynamique est meilleure que la modélisation statique dans le cas monophonique (DB1), mais pas dans le cas polyphonique (DB2): cela signifierait que la polyphonie ruine les efforts de modélisation temporelle même dans le cadre restreint de notes individuelles. Finalement, si les algorithmes dynamiques surpassent les algorithmes statiques sur DB1 et DB2, mais pas dans le cas des textures de plusieurs notes (comme nous l'avons établi au chapitre prédédent), alors l'hypothèse H3 sera preférée.

Nous testons sur chaque base de données un série de 17 variantes algorithmiques, dont 3 modèles dynamiques, basés sur la programmation dynamique. La précision des modèles est calculée selon le même mode que précédemment (*R*-précision). Les deux bases de données DB1 et DB2 comprennent le même nombre de signaux (700) et le même nombre de classes (16), ce qui crée des tâches de complexité comparable.

5.4 Résultats

La comparaison des résultats sur les deux bases de données montre que les algorithmes dynamiques sont plus de *10% plus précis* que leurs homologues statiques sur la base monophonique (DB1). Ceci confirme les conclusions des précédentes études sur la classification de sons instrumentaux. Les meilleurs algorithmes statiques sur DB1 utilisent des modèles assez compliqués (GMM avec plusieurs composantes gaussiennes), et de larges concaténations d'attributs. Toutefois, les mêmes algorithmes dynamiques sont plus de *10% moins précis* que les algorithmes statiques sur la base polyphonique DB2. Les meilleurs résultats sur DB2 sont obtenus pour de très simples algorithmes,

comme la comparaison euclidienne de la moyenne des vecteurs MFCC.

Cette expérience montre donc que la difficulté à modéliser la dynamique des séquences d'attributs instantanés est dûe au caractère polyphonique des données (H2), et non à leur structure temporelle (en notes, phrases, refrains, etc.). Ce résultat suggére que le codage horizontal des trames audio, qui ne prend pas en compte les différentes sources et leur synchronisation verticale, est une représentation peu efficace pour la musique polyphonique, et peu plausible cognitivement.

Chapitre 6: Des hubs et de leurs propriétés

Une des conclusions les plus importantes de l'**expérience 1** est que la classe d'algorithmes étudiée ici crée de faux-positifs qui sont presque toujours les mêmes chansons, indépendamment de la requête de départ. Entre d'autres termes, il existe pour ces algorithmes des chansons anormalement proches de toutes les autres chansons: des *hubs*.

6.1 Définition

On définit comme un "hub" une chanson qui apparaît fréquemment comme faux-positif pour une mesure de similarité donnée. Cela implique à la fois que

1. un hub apparaît dans les plus proches voisins de nombreuses chansons de la base de données
2. la plupart de ces apparitions ne révélent pas une similitude perceptuelle avec la chanson-requête

Chacune de ces conditions n'est pas suffisante en soi. Par exemple, une chanson peut apparaître souvent dans les voisins de la plupart des requêtes, mais de façon correcte perceptivement: cela révéle que cette chanson est une sorte de centre de masse perceptif pour la base de données. Par exemple, *A Hard Day's Night* est peut-être une chanson proche perceptivement de toutes les autres chansons des *Beatles*. Cela n'en fait pas un hub. Réciproquement, une chanson donnée peut être un faux-positif pour une requête donnée, mais pas pour un grand nombre d'entre elles. Par exemple, *A Hard Day's Night* peut être estimée comme proche d'une sonate pour piano de *Beethoven* par une mesure donnée: c'est un bug, car les deux chansons n'ont aucune similitude perceptive. Toutefois, cela ne fait pas de *A Hard Day's Night* un hub, s'il n'est associé anormalement qu'à cet exemple précis.

6.2 Importance du phénomène

Ce phénomène de hubs rappelle curieusement des observations déjà faites dans d'autres domaines que la musique, notamment en identification biométrique de locuteurs ou d'empreintes digitales.

Le rapprochement est d'autant plus intéressant à faire que ces technologies emploient typiquement les même techniques algorithmiques que celles que nous étudions ici. Cela incite à penser que ce phénomène est une caractéristique importante de ce type d'algorithmes, et que nos conclusions s'étendent donc au delà du problème spécifique de la similitude timbrale de signaux musicaux.

6.3 Deux mesures de hubness

Nous proposons deux mesures du degré de hubness pour une chanson:

- le nombre d'occurrences: mesure le nombre de fois qu'une chanson donnée apparaît dans les *k* plus proches voisins de toutes les autres chansons dans la base de données. On montre que la moyenne des nombres d'occurrences de toutes les chansons d'une base est égale à *k*. Si une chanson apparaît notablemment plus que la moyenne, elle peut être considérée comme un hub. Dans la base de test utilisée dans l'**expérience 1**, une chanson, *Joni Mitchell - Don Juan's Reckless Daughter*, apparaît 57 fois dans les 10 plus proches voisins des autres chansons de la base, soit prés de 6 fois la valeur théorique.

- l'angle aux voisins: Un hub peut également être défini comme une chanson proche de chansons qui sont très éloignées entre elles. Le rapport entre la distance aux voisins d'une part, et la distance entre voisins d'autre part, peut s'exprimer comme un angle, dont le cosinus se calcule par une simple trigonométrie. On montre que la valeur moyenne des angles aux voisins d'une base donnée est de 60 degrés. Tout chanson dont l'angle aux voisins moyen est notablement supérieur à 60° peut donc être considérée comme un hub.

6.4 Les hubs forment une distribution à invariance d'échelle

La distribution des nombre d'occurrences dans les 100 plus proches voisins sur une base de chansons de 15,000 mp3 montre un profil de loi de puissance. L'immense majorité des chansons ont autour d'une centaine d'occurrences (ce qui correspond à la valeur moyenne), toutefois, un très petit nombre de chansons ont un nombre d'occurrence extrêmement grand, jusqu'à 4000, ce qui est

plus de 400 fois la valeur attendue.

6.5 Les hubs sont une conséquence de la modélisation de l'agglomération des attributs, pas des attributs eux-mêmes (Expérience 3)

On montre qu'une base de données de trames individuelles, modélisées avec des MFCCs, et comparées par simple distance euclidienne, ne génère pas de hubs. Pour une base de 15000 trames, le nombre d'occurrences maximum observé est de 400, ce qui est plus de 10 fois inférieur au nombre observé dans une base composée du même nombre de séquences de trames correspondant à des morceaux entiers. Cela indique que les hubs ne sont pas une conséquence d'un mauvais comportement des attributs individuels, mais de la modélisation de leur agglomération à long-terme.

6.6 Les hubs apparaissent pour tous les algorithmes (Expérience 4)

La comparaison de plusieurs variantes algorithmiques implémentées dans le cadre de l'expérience 1 montre que tous les algorithmes génèrent des hubs. En particulier, les modèles non-paramétriques (histogrammes) génèrent autant (sinon plus) de hubs que leurs homologues paramétriques (GMMs). Similairement, les variantes "à dynamique" comme les modèles de Markov cachés ou les coefficients dérivés génèrent plus de hubs que les simples GMMs.

6.7 Le fait d'être un hub n'est pas intrinsèque à une chanson donnée (Expérience 5)

Le degré de hubness des chansons de la base de test ne sont pas fortement corrélés pour différentes mesures de similitudes. Cela indique que les algorithmes distribue les hubs différemment sur la base de données, et que le fait d'être un hub n'est pas intrinsèque à une chanson donnée, mais dépend de l'algorithme utilisé. Les hubs générés par les variantes dynamiques (HMM, delta, accélération) sont plus corrélés entre eux qu'avec les hubs générés par les algorithmes statiques.

6.8 Les hubs ne sont pas la conséquences de modèles peu discriminants (Expérience 6)

Une hypothèse séduisante pour expliquer l'apparition de hubs propose qu'ils soient le résultat de modèles peu discriminants, et donc prompts à expliquer des données très hétéroclites. De tels modèles peuvent correspondre par ex. aux chansons dont la modélisation statistique emploie de fortes variances, ou pour des algorithmes statiques qui considèrent toutes les permutations des trames de la chanson originale comme rigoureusement identiques. On montre toutefois que les hubs ne sont pas corrélés aux chansons dont les modèles ont une forte variance, et que les algorithmes statiques (en particulier paramétriques) ne créent pas moins de hubs que leurs homologues dynamiques plus contraints.

6.9 Le fait d'être un hub peut être localisé à certaines trames seulement (Expérience 7)

L'importance d'un hub donné n'est pas une propriété émergeant globalement de la distribution de ses attributs instantanés, mais peut être attribuée à certaines parties de cette distribution. On montre que les 5% de trames les moins représentative statistiquement (cad correspondant aux composantes gaussiennes de poids les plus faibles dans les GMM) sont extrêmement importantes à considérer, et que leur suppression augmente radicalement le nombre de hubs dans la base de données. Si l'on continue d'homogénéiser les GMMs en supprimant les composantes les moins statistiquement importantes, le nombre de hubs décroît à nouveau, jusqu'à un minimum local suite à la suppression des 40% de trames les moins importantes. Cela indique qu'il existe une population de trames dans la région de poids statistique $[60\%, 95\%]$ qui sont très peu discriminantes et qui sont en majorité responsables de l'apparition de hubs.

Ce phénomène traduit une situation générale dans laquelle les observations instantanées n'ont pas toutes la même importance perceptuelle, qui en particulier ne dépend pas de leur saillance statistique par rapport à leur distribution à long-terme: les trames les plus informantes pour discriminer

des textures polyphoniques sont également les moins représentatives d'un point de vue statistique (les 5% inférieurs), alors qu'une grande population de trames (plus d'un tiers) a un impact très négatif sur la modélisation, en étant "proches de tout le reste".

6.10 Les hubs n'apparaissent pas pour tout type de données (Expérience 8)

Nous établissons enfin que l'apparition de hubs ne dépend pas que de l'algorithme utilisé, mais aussi du type de signaux modélisés. Notamment, les mêmes mesures appliquées à des signaux audio correspondant à des ambiances environnementales (enregistrements de parc, boulevards, rue piétonnes, etc.) ont des précisions jusqu'à 30% supérieures à celles obtenues pour la musique polyphonique, et n'engendrent pas de hubs. On montre qu'une distinction importante entre ces 2 classes de signaux "à hubs" et "sans hubs" peut être formulée en termes d'homogénéité temporelle et statistique. Notamment, les textures environnementales sont bien modélisées même en ne considérant que de courts extraits: seulement 10% de précision perdue (relatif) en passant de signaux de 3 minutes à 10 secondes. Au contraire, la modélisation de musique polyphonique nécéssite beaucoup plus de données: plus de 60% de précision perdue (relatif) en passant de 3 minutes à 10 secondes.

Chapitre 7: L'ancrage timbral des jugements sémantiques

7.1 Inférer des descriptions haut-niveau avec le timbre (Expérience 9)

Nous examinons ici la validité d'utiliser un modèle de timbre pour extraire des métadonnées musicales haut-niveau. Notre étude repose sur une base de données interne à SONY CSL, qui contient 4936 chansons, chacune décrite avec un ensemble de 801 attributs booléens (par ex. "Language English = true"). Ces attributs sont regroupés en 18 catégories, qui décrivent des aspects acoustiques du son musical (par ex. instrument, dynamique), mais aussi des descriptions plus éditoriales (language, pays), culturelles ou subjective (genre, émotion, situation). Ces attributs sont un éventail représentatif des métadonnées typiquement modélisées dans la communauté *MIR*.

Nous étudions la précision d'un mécanisme d'inférence de ces attributs reposant uniquement sur la mesure de similitude timbrale développée dans cette étude. L'algorithme de classification est inspiré des k-plus-proches-voisins: une chanson est classifiée comme par ex. "hard rock" si une grande partie de ses plus proches voisins au sens de la similitude timbrale sont eux aussi classifiés comme tel. Plus précisément, on apprend la densité de probabilité du nombre d'occurrences de chansons d'une catégorie C dans les plus proches voisins d'une chanson dans le cas où cette chanson appartient à C d'une part, et n'y appartient pas d'autre part. La décision pour une chanson s non encore classifiée d'appartenir ou non à C se fait ensuite par maximum de vraisemblance (*maximum likelihood*) en comptant le nombre de chansons appartenant à C dans les plus proches voisins de s.

Les résultats montrent que certains attributs (Style Techno, Genre Metal...) sont extrêmement bien estimés avec la similitude timbrale, avec des précisions parfois supérieures à 95%. Toutefois, ces attributs bien corrélés au timbre sont extrêmement rares: seulement 6 % des attributs sont estimés avec plus de 80% de précision, et plus de la moitié des attributs obtiennent moins de 65% (ce qui est à peine meilleur qu'une décision binaire faite aléatoirement). Cela indique que très peu de descriptions musicales haut-niveau ont une définition consensuelle en termes d'un timbre prototypique.

D'autre part, on observe que tous les taxons d'une catégorie donnée n'ont pas le même com-

portement. Les genres “unplugged” ou “hard rock” sont de forts corrélats du timbre, car ils correspondent à des signaux très homogènes et typés. Toutefois, les genres “jingle” ou “electronica” sont de piètres corrélats timbraux, car ce sont des groupes très hétérogènes, et mal définis en termes de couleur sonore. Cela entre en contradiction avec la stratégie habituellement utilisée en *MIR* d’utiliser le même espace de décision pour discriminer tous les taxons d’une catégorie donnée.

Finalement, il apparaît que beaucoup d’attributs de la catégorie “instrument” sont eux-aussi mal modélisés par la similitude timbrale. Cela montre que les jugements humains en matière de timbre polyphonique ne sont pas le résultat immédiat de perceptions bas-niveau, mais plutôt de raisonnements cognitifs évolués, dépendant par exemple du contexte d’écoute et de la culture de l’auditeur.

7.2 Le rôle du contexte et des corrélations haut-niveau (Expérience 10)

Nous montrons ici qu’il existe néanmoins de très importantes corrélations statistiques entre les attributs, indépendamment de leur ancrage sur le timbre, qui révèlent la possibilité d’un important effet de contexte lors des processus de décision. Pour ce faire, nous mesurons l’indépendance statistique entre tous les couples d’attributs de la base, avec le test du khi-carré (*Pearson’s* χ^2). Parmi les couples d’attributs les moins indépendants statistiquement, on observe plusieurs types de corrélation:

- Tautologies: Un certain nombre de corrélations traduisent des associations triviales de mots lexicalement proches, comme “TextCategory Christmas” et “Situation Christmas”.

- Sur-apprentissage: des corrélations dûes à l’existence d’un tout petit nombre, non représentatif, de chansons présentant une combinaison particulière d’attributs, par ex. “Toutes les chansons parlant de bicyclette utilise de la guimbarde”. Ce genre de règles non pertinentes sont faciles à éviter en pratique, par filtrage ou validation croisée.

- Exclusions: des corrélations négatives, d’ordre logique: une chanson ne peut être à la fois de tempo “constant” *et* “varié”, ou “vocale” *et* “instrumentale”.

- Associations sémantiques: des corrélations intrinsèques aux termes utilisés, de l'ordre de la psycholinguistique, et qui n'ont que peu à voir avec leur utilisation dans un cadre musical. Par ex. une chanson "puissante" (*powerful*) est une chanson "forte" (*strong*), une chanson qui parle d'"anniversaire" est à utiliser lors d'"occasions spéciales".

- Connaissances musicales: des corrélations qui ne sont pas des propriétés intrinsèques des termes employés, mais des propriétés extrinsèques propres au domaine musical, par ex. des connaissances historiques ou culturelles entre genres ("Rap" et "Hip hop"), genres et pays ("bossa nova" et "Brésil"), genres et instruments ("hip hop" et *"spoken vocals"*), genres et époques ("rag time" et "1930's"), et même des aspects plus subjectifs liés à la perception du timbre ("la flute crée un son "chaleureux" (*warm*)").

Ces résultats suggèrent que beaucoup de métadonnées musicales haut-niveau peuvent effectivement être ancrées sur perception du timbre, mais uniquement en exploitant ces corrélations haut-niveau avec les meilleurs corrélats timbraux.

7.3 Un modèle hybride d'ancrage de descriptions haut-niveau

Nous présentons ici un algorithme qui réalise ce projet, en utilisant une mesure de similitude timbrale comme amorçage (*bootstrap*) pour estimer quelques attributs bien corrélés au timbre, puis en utilisant des arbres de décision pour estimer d'autres attributs non corrélés directement, mais pour lesquels il existe des corrélations haut-niveau avec l'ensemble des attributs amorcés timbralement.

Les résultats, préliminaires sur 45 attributs choisis aléatoirement parmi les 801 disponibles, montre que l'utilisation de corrélations haut-niveau d'ordre culturel permet d'améliorer la précision des estimations faites uniquement avec le timbre. Dix attributs voient la précision de leur estimation augmenter de plus de 10%, et 15 attributs ont une précision finale supérieure à 75%. En particulier, deux attributs de la catégorie "instrument" ("guitare" et "choeur"), qui sont de piètres corrélats timbraux (60%), sont améliorés en utilisant des corrélations avec d'autre attributs, ce qui illustre la pertinence de notre approche.

Conclusion : Vers des modèles cognitifs

Les expériences décrites dans cette étude illustrent un certain nombre de contradictions entre les modèles computationnels du timbre polyphonique et sa perception humaine. Notamment, nous avons constaté que les observations instantanées n'ont pas toutes la même importance perceptuelle, qui en particulier ne dépend pas de leur saillance statistique par rapport à leur distribution à long-terme. Cet effet est possiblement à l'origine des *hubs* observés sur tous les algorithmes étudiés ici. L'approche "GMM de MFCCs" crée une représentation de l'objet sonore qui est essentiellement *extensionelle*. Cela semble être un modèle suffisant pour les textures environnementales, qui d'une certaine façon, sont définies exclusivement par ce qu'elle sont, statistiquement. Toutefois, nos expériences montrent que la perception de la musique, et particulièrement de textures polyphoniques extraites de morceaux connus, est une construction principalement *intensionnelle*. En particulier, certains timbres sont plus saillants à la perception que d'autres: la première chose que nous percevons d'une chanson d'un artiste donné est souvent le timbre particulier de sa voix, même si celle-ci n'est pas prédominante d'une point de vue statistique, et non l'accompagnement à base de guitare, piano ou autres.

En d'autres termes, nous "entendons" quotidiennement dans la musique polyphonique des choses qui ne sont pourtant pas présentes de façon significative (statistiquement) dans le signal sonore. La musique que nous *entendons* être du piano est surtout de la musique que nous nous *attendons* à être du piano. Ces paradoxes statistico-perceptifs, dont l'existence devrait maintenant être établie de façon rigoureuse, expliquent en grande partie le désaccord entre les modèles algorithmiques étudiés ici et la perception humaine.

Part IV

Appendices

Appendix A

Composition of the test database

In the context of large, systematic evaluation of very many algorithm variants, the subjective evaluations of each possible similarity variant by human subjects, as was done for the psychophysical investigations described in Chapter 2, seems unpractical. However, objective evaluation is also problematic, because of the choice of a ground truth to compare the measure to. Several authors have proposed solutions for this: Logan and Salomon (2001) consider as a good match a song which is from the "same album", "same artist", "same genre" as the seed song. Pampalk et al. (2003) also propose to use "styles" (e.g. Third Wave ska revival) and "tones" (e.g. energetic) categories from the *All Music Guide*[*]. Berenzweig et al. (2003) push the quest for ground truth one step further by mining the web to collect human similarity ratings.

For this study, we have constructed a test database of 350 song items, as an extract from the Cuidado database (Pachet et al., 2004), which currently has 15,460 mp3 files. It is composed of 37 clusters of songs by the same artist, which were refined by hand to satisfy 3 additional criteria:

- First, clusters are chosen so they are as distant as possible from one another. This is realized e.g. by choosing artists of very different genres, that span the whole space of music available in the database (from *Beethoven* to *The Clash*).

[*]www.allmusic.com

- Second, artists and songs are chosen in order to have clusters that are "timbrally" consistent (all songs in each cluster sound the same). This is realized typically by choosing sets of songs for a given artist that all have a distinguished "sound" signature of a given period of activity of the artist. For instance, all songs from the *Beatles* cluster come from their 1963-1964 albums (*Please Please Me*, *With the Beatles*, *A Hard Day's Night* and *Beatles for Sale*), released on *Capitol*.

- Finally, we only select songs that are timbrally homogeneous, i.e. there is no big texture change within each song. This is to account for the fact that we only compute and compare one timbre model per song, which "merges" all the textures found in the sound. In the case of more heterogeneous songs (e.g. *Queen - Bohemian rhapsody*), higher-level models such as a segmentation step could increase the accuracy of the measure, but such techniques are not considered in this study (see for instance Foote (2000)).

This test database is constructed so that nearest neighbors of a given song should optimally belong to the same cluster as the seed song. Table A.1 lists the clusters in the database.

Table A.1: Composition of the test database. The “descriptions” were taken from the AMG (www.allmusic.com).

Artist	Description	Size
ALL SAINTS	Dance Pop	9
APHEX TWIN	Techno	4
BEATLES	British Pop	8
BEETHOVEN	Classical Romantic	5
BRYAN ADAMS	Pop Rock	8
FRANCIS CABREL	French Pop	7
CAT POWER	Indie Rock	5
CHARLIE PATTON	Delta Blues	10
THE CLASH	Punk Rock	21
VARIOUS ARTISTS	West Coast Jazz	14
DD BRIDGEWATER	Jazz Singer Trio	12
BOB DYLAN	Folk	13
ELTON JOHN	Piano Pop	5
FREHEL	French Prewar Singer	8
GARY MOORE	Blues Rock	9
GILBERTO GIL	Brazilian Pop	15
JIMI HENDRIX	Rock	7
JOAO GILBERTO	Jazz Bossa	8
JONI MITCHELL	Folk Jazz	9
KIMMO POHJONEN	World Accordion	5
MARDI GRAS BB	Big Band Blues	7
MILFORD GRAVES	Jazz Drum Solo	4
VARIOUS	“Musette” Accordion	12
PAT METHENY	Guitar Fusion	6
VARIOUS ARTISTS	Jazz Piano	15
PUBLIC ENEMY	Hardcore Rap	8
QUINCY JONES	Latin Jazz	9
RASTA BIGOUD	Reggae	7
RAY CHARLES	Jazz Singer	8
RHODA SCOTT	Organ Jazz	10
ROBERT JOHNSON	Delta Blues	14
RUN DMC	Hardcore Rap	11
FRANK SINATRA	Jazz Crooner	13
SUGAR RAY	Funk Metal	13
TAKE 6	Acapella Gospel	10
TRIO ESPERANCA	Acapella Brasilian	12
VOCAL SAMPLING	Acapella Cuban	13

Appendix B

Experiment 1 - Details

This appendix gives full details about **Experiment 1** (Chapter 5), in which we explore the space of polyphonic timbre models by applying transformations (so-called *design patterns*) to the prototypical algorithm described in Chapter 4. The evaluation methodology follows the description made in Chapter 5. Notably, we report results using scores of *R*-precision (the precision measured after *R* documents have been retrieved, where *R* is the number of relevant documents) averaged over all queries in the test database. The complete analysis of these results can be found in Chapter 5.

B.1 Tuning feature and model parameters (patterns 3.2.1 and 3.2.2)

As a first evaluation, we wish to find the best set of parameters for the original algorithm ("GMMs of MFCCs"). We explore the space constituted by the following parameters :

- Signal Sample Rate (SR): The sample rate of the music signal. In the classic framework of Fourier analysis, a given sample rate SR invalidates all frequencies above $\frac{SR}{2}$. Humans can hear vibrations ranging from about 20 Hz to approximately 20 kHz, so sampling that doesn't extend this far will have a detrimental effect on the resultant quality. The red book standard for audio CD is $44,1$kHz, and high-quality digital recording equipments typically offer sample rates up to 96 or 192kHz. The original value used in Aucouturier and Pachet

(2002b) is 11KHz. This value was chosen mainly to reduce the CPU runtime of the prototype Matlab implementations.

- Number of MFCCs (N): The number of the MFCCs extracted from each frame of data. The more MFCCs, the more precise the approximation of the signal's spectrum, which also means more variability on the data. As we are only interested in the spectral envelopes, not in the finer, faster details like pitch, a large number may not be appropriate. The original value used in Aucouturier and Pachet (2002b) is 8.

- Number of Components (M): The number of gaussian components used in the GMM to model the MFCCs. The more components, the better precision on the model. However, depending on the dimensionality of the data (i.e. the number of MFCCs), more precise models may be underestimated. The original value is 3.

- Distance Sample Rate (DSR): The number of points used to sample from the GMMs in order to estimate the likelihood of one model given another. The more points, the more precision on the distance, but this increases the required CPU time linearly.

- Window Size :The size of the frames on which we compute the MFCCs.

As this 5-dim space is too big to explore completely, we make the hypothesis that the influence of SR, DSR, and Window Size are both independent of the influence of N and M. However, it is clear from the start that N and M are linked: there is an optimal balance to be found between high dimensionality and high precision of the modeling (*curse of dimensionality*).

B.1.1 influence of SR

To evaluate SR, we fix N, M and DSR to their default values used in Aucouturier and Pachet (2002b) (8,3 and 2000 resp.). In Table B.1, we see that the SR has a positive influence on the precision. This is probably due to the increased bandwidth of the higher definition signals, which enables the algorithm to use higher frequency components than with low SR.

Table B.1: Influence of signal's sample rate

SR	*R*-Precision
11kHz	0.488
22kHz	0.502
44kHz	0.521

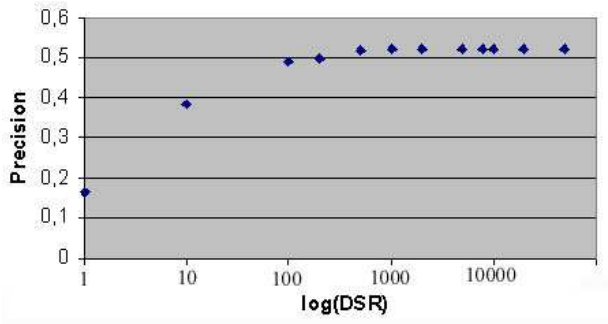

Figure B.1: Influence of the distance sample rate

B.1.2 influence of DSR

To evaluate DSR, we fix N = 8, M=3 and SR= 44KHz. In Figure B.1, we see that the DSR has a positive influence on the precision when it increases from 1 to 1000, and that further increase has little if any influence. Further tests show that the optimal DSR does not depend on either N or M.

In Appendix D.3, we use this property and the resulting tradeoff between precision and cputime to greatly optimize the time needed to compute nearest neighbors.

B.1.3 influence of N,M

To explore the influence of N and M, we make a complete exploration of the associated 2-D space, with N varying from 10 to 50 by steps of 10 and M from 10 to 100 by steps of 10. These boundaries result from preliminary tests (moving N while M=3, and moving M while N=8) showing that both default values N=8 and M=3 are not optimal, and that the optimal (N,M) was well above (10,10). Figure B.2 shows the results of the complete exploration of the (N,M) space.

We can see that too many MFCCs ($N \geq 20$) hurt the precision. When N increases, we start to

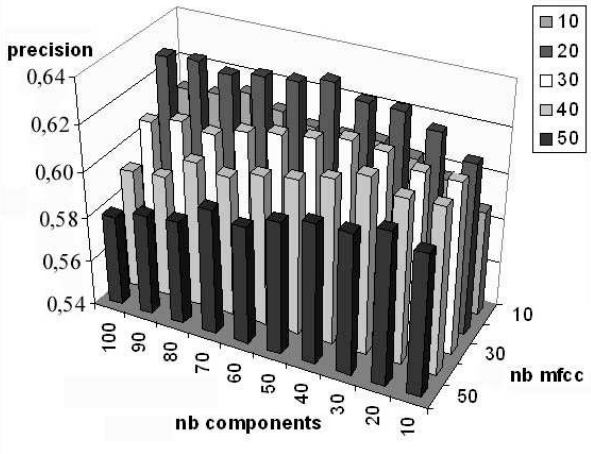

Figure B.2: Influence of the number of MFCCs and the number of components

take greater account of the spectrum's fast variations, which are correlated with pitch. This creates unwanted variability in the data, as we want similar timbres with different pitch to be matched nevertheless.

We also notice that increasing the number of components at fixed N, and increasing N at fixed M is eventually detrimental to the precision as well. This illustrates the curse of dimensionality mentioned above. The best precision $p = 0.63$ is obtained for 20 MFCCs and 50 components. We can also note that the number of MFCCs is a more critical factor than the number of Gaussian components : $\forall M, N \neq 20, p(N_0 = 20, M) \geq p(N, M)$. This means we can decrease M to values smaller than optimum without much hurting the precision, which is an interesting point as the computational cost of comparing models depends linearly on M.

B.1.4 influence of Windows Size

To evaluate the influence of the window size used to segment the waveforms, we fix N = 20, M=50, SR=44 KHz and DSR = 2000. In Figure B.3, we see that the window size has a small positive influence on the precision when it increases from 10 ms to 30ms, but that further increase up to 1 second has a negative effect. This behaviour results from the fact that MFCCs are only meaningful on stationary frames (larger frames may include more transients and variations) and that larger

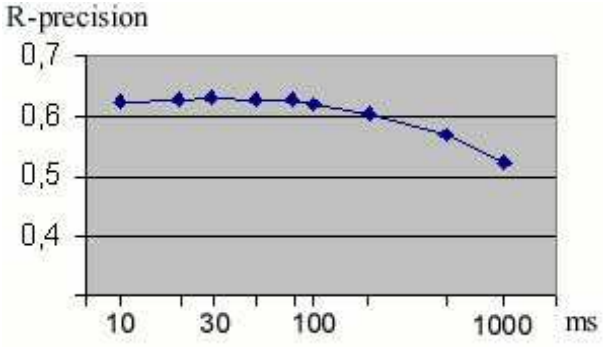

Figure B.3: Influence of the windows size

frames means less data available for the training, which decreases the precision of the models.

B.2 Alternative distance measure (pattern 3.2.4)

Some authors (Logan and Salomon (2001); Berenzweig et al. (2003)) propose to compare the GMMs using the Earth Mover's distance (EMD), a distance measure meant to compare histograms with disparate bins (Rubner et al. (1998)). EMD computes a general distance between GMMs by combining individual distances between gaussian components. It is defined as the minimum amount of work needed to change one set of gaussian components into the other. The notion of "work" is based on a user-defined distance between individual components. The implementation of EMD is based on a classical solution to the *transportation problem*.

The computation of the distance between 2 GMMs I and J is defined as a minimum flow problem between a set of suppliers (the gaussian components of GMM I) and a set of consumers (the components of GMM J). Suppliers have an amount of supply w_i, which is the mixture coefficient of the corresponding gaussian component (see Section3.1). Similarly, the mixture coefficients of consumers define their capacity. Shipping a unit of supply from a supplier w_i to a consumer w_j is associated with a cost c_{ij}. We want to find a set of flows f_{ij} (i.e. the amount of supply shipping from

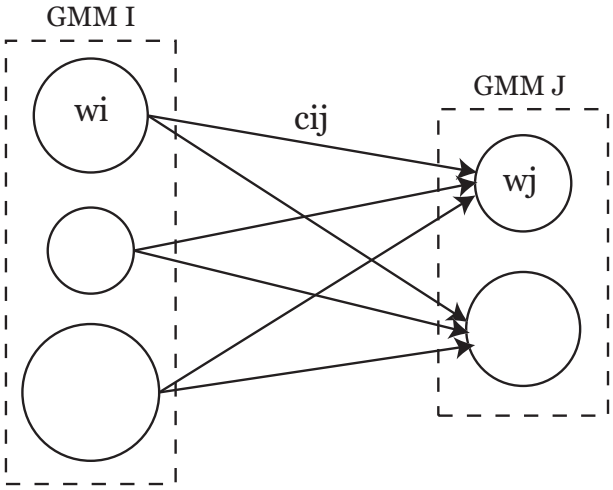

Figure B.4: Earth Mover's Distance between 2 Gaussian Mixture Models, viewed as a transportation problem between suppliers (components of GMM *I*) and consumers (components of GMM *J*).

each supplier to each consumer) that minimizes the overall cost:

$$cost = \sum_i \sum_j c_{ij} f_{ij} \tag{B.1}$$

This is a linear programming problem, subject to the following constraints :

$$f_{ij} \geq 0 \quad \forall i, j \tag{B.2}$$

$$\sum_i f_{ij} = w_j \quad \forall j \tag{B.3}$$

$$\sum_j f_{ij} \leq w_i \quad \forall i \tag{B.4}$$

Constraint B.3 ensures that supply only flows from suppliers to consumers. Constraint B.4 ensures that consumers are filled to their full capacity, while Constraint B.4 forces suppliers to send no more supply than their initial amount. Note that the constraints are compatible since by construction $\sum_i w_i = \sum_j w_j = 1$. A simple solution to tha above system can be found by the classic simplex method (Press et al., 1986).

The costs c_{ij} between individual gaussian components can be defined e.g. as the classic Kullback

Table B.2: Distance function

Distance	*R*-Precision
EMD-KL	0.477
EMD-MA	0.406
DSR=2000	0.488

Leibler distance (as seen in Section 3.1), or the related Mahalanobis distance:

$$D_M(x) = \sqrt{(x-\mu)^T \Sigma^{-1} (x-\mu)} \tag{B.5}$$

To evaluate the EMD against our sampling scheme using DSR, we fix N = 8, M=3 and SR= 11KHz. We compare EMD with Kullback-Leibler (KL), EMD with Mahalanobis (MA) and sampling with DSR=2000. In Table In B.2, we see that EMD with Mahalanobis distance performs worst, and that EMD with Kullback Leibler and sampling are equivalent (with a slight advantage to sampling). The difference between MA and KL is probably due to the fact that MA takes less account of covariance differences between components (2 gaussian components having same means and different covariance matrices have a zero Mahalanobis distance).

B.3 Feature Composition (pattern 3.2.5)

B.3.1 Processing commonly used in Speech Recognition

MFCCs are a very common front-end used in the Speech Recognition community (Rabiner and Juang, 1993), and a variety of pre and post-processing has been tried and evaluated for speech applications. Here we examine the influence of 6 common operations :

- ZMeanSource: The DC mean is removed from the source waveform before doing the actual signal analysis. This is used in speech to remove the effects of A-D conversion.

- Pre-emphasis: It is common practice to pre-emphasize the signal by applying the first order difference equation :

$$s'_n = s_n - k s_{n-1} \tag{B.6}$$

to the samples s_n in each window, with k a preemphasis coefficients, $0 < k < 1$. Pre-emphasis is used in speech to reduce the effects of the glottal pulses and radiation impedance and to focus on the spectral properties of the vocal tract.

- Dither: Certain kind of waveform data can cause numerical problems with certain coding schemes (*finite wordlength effects*). adding a small amount of noise to the signal can solve this. The noise is added to the samples using :

$$s'_n = s_n + q\mathcal{RND} \tag{B.7}$$

where $\mathcal{RND}$ is a uniformly distributed normalized random value and q is a scaling factor.

- Liftering: Higher order MFCCs are usually quite small, and this results in a wide range of variances from low to high order. this may cause problems in distribution modeling. Therefore it is common practice in speech to rescale the coefficients to have similar magnitude. This is done by filtering in the cepstrum domain (L*i*F*tering*) according to :

$$c'_n = (1 + \frac{L}{2} \sin \frac{\pi n}{L} c_n) \tag{B.8}$$

where L is a liftering parameter.

- Cepstral mean compensation (CMC): The effect of adding a transmission channel on the source signal is to multiply the spectrum of the source by a channel transfer function. In the cepstral log domain, this multiplication becomes an addition which can be removed by subtracting the cepstral mean.

- 0'th order coefficient: The 0'th cepstral parameter C_0 can be appended to the c_n. It is correlated with the signal's log energy :

$$E = \log \sum_{n=1}^{N} s_n^2 \tag{B.9}$$

Table B.3 shows the results on the test database. We notice that subtracting the cepstral mean severely degrade the performance. Pre-emphasis and Dither have little effect compared to the original MFCCs. Nevertheless, liftering, normalizing the original signal and appending the 0'th coefficient all improves the precision of the measure.

We should note here that the finding that including c0 slightly improves the performance is at odds to some of the results reported in Berenzweig et al. (2003). In any case, the overall influence (either positive here or negative elsewhere) of this variant is small (a few percent). We further discuss these results in Chapter 6.

B.3.2 Spectral Contrast

In Jiang et al. (2002), the authors propose a simple extension of the MFCC algorithm to better account for music signals. Their observation is that the MFCC computation averages the spectrum in each sub-band, and thus reflects the average spectral characteristics. However, very different spectra can have the same average spectral characteristics. Notably, it is important to also keep track of the relative spectral distribution of peaks (related to harmonic components) and valleys (related to noise). Therefore, they extend the MFCC algorithm to not only compute the average spectrum in each band (or rather the spectral peak), but also a correlate of the variance, the *spectral contrast* (namely the amplitude between the spectral peaks and valleys in each subband). This modifies the algorithm to output 2 coefficients (instead of one) for each Mel subband. Additionally, in the algorithm published in Jiang et al. (2002), the authors replace the Mel filterbank traditionally used in MFCC analysis by an octave-scale filterbank (C_0-C_1, C_1-C_2, etc.), which is assumed to be more suitable for music. They also decorrelate the spectral contrast coefficients using the optimal

Table B.3: Influence of Feature Variants

Variant	*R*-Precision
Acceleration $\Theta = 50$	0.179
SpectralMP7	0.514
Delta $\Theta = 50$	0.522
Cepstral Mean Compensation	0.525
SpectralMP7	0.573
Delta $\Theta = 10$	0.60
Acceleration $\Theta = 10$	0.610
Delta $\Theta = 2$	0.624
Delta $\Theta = 5$	0.625
Acceleration $\Theta = 5$	0.625
Pre Emphasis $k = 0.97$	0.628
Acceleration $\Theta = 1$	0.628
Original MFCC	0.629
Dither $q = 5\%$	0.629
Lifter $L = 22$	0.630
Delta $\Theta = 1$	0.631
ZMeanSource	0.631
Acceleration $\Theta = 2$	0.631
0’th coefficient	0.652
Best 3	0.605
Best 4	0.609

Table B.4: Influence of Spectral Contrast

Implementation	*R*-Precision
SC1	0.640
SC2	0.656
SFN	0.636
standard MFCC	0.629

Karhunen-Loeve transform.

We have implemented and evaluated two variants of Spectral Contrast here. For convenience, both variants use the MFCC Mel filterbank instead of the authors' Octave filters, and use the MFCC's Discrete Cosine Transform to approximate the K-L Transform. This has the advantage of being data independent, and thus better adapted to the implementation of a similarity task, where one wish to be able to assess the similarity between any duplet of song without first having to consider the whole available corpus (as opposed to the authors' supervised classification task, where the KL can be trained on the total data to be classified). Moreover, it has already been reported that the DCT was a satisfying approximation of the K-L transform in the case of music signals (Logan (2000). In the first implementation (SC1), the $2N_{chan}$ coefficients (where N_{chan} is the number of subbands in the filterbank) are all appended in one block, and reduced to N cepstrum coefficients using the dct. In the second implementation, both streams of data (the N_{chan} peaks and the N_{chan} Spectral Contrast) are decorrelated separately with the DCT, resulting in $2N$ cepstral coefficients, as if we used e.g. delta coefficients.

We see that both front-ends perform about 1% better than standard MFCCs, and that the $2N$ implementation performs best. For further improvement, Spectral Contrast could be combined with traditional Pre/Post Processing as seen above.

B.4 Feature Equivalence (pattern 3.2.3)

We have tried replacing/appending to the MFCC feature set a set of MPEG7-standardized spectral descriptors based on moments of the spectrum, as described in Section 3.2.3: `SpectralCentroid`,

`SpectralSpread`, `SpectralKurtosis`, `SpectralSkewness`, `SpectralFlatness`, `SpectralRolloff`, `SpectralFlux`. The resulting feature vector (dim 7) was modelled with $M = 20$ Gaussian mixture models, both *as is* and concatenated with a vector of 20 MFCCs. Table B.3 shows the performance of both approaches. The MP7-only feature set performs significantly worse than the best MFCC settings ($N = 20$, $M = 50$), but also than a set of MFCCs with equivalent dimensions ($N = 8, M = 20$) (Figure B.2). The combined feature vector of MFCC and MP7 descriptors also performs slightly worse than both the optimal MFCC-only settings.

B.5 Modelling dynamics (pattern 3.2.7)

B.5.1 Delta and Acceleration Coefficients

It is known since Furui (1986) that the performance of a speech recognition system can be greatly enhanced by adding time derivatives to the basic static parameters. Delta Coefficients are computed using the following formula :

$$d_t = \frac{\sum_{\theta=1}^{\Theta} \theta(c_{t+\theta} - c_{t-\theta})}{2\sum_{\theta=1}^{\Theta} \theta^2} \tag{B.10}$$

where d_t is a delta coefficient at time t, computed using a time window Θ. The same formula can be applied to the delta coefficients to obtain the acceleration coefficients.

Table B.3 shows the performance of adding delta and/or acceleration coefficients to the original MFCC dataset, for various values of Θ. We notice that computing delta and acceleration coefficients for large time windows severely degrade the performance. However, appending short-term delta and acceleration coefficients improves the precision of the measure.

We have tried to combine the best operations in Table B.3 (which is referred to as “Best 3” and “Best 4”), however this does not further improve the precision. We should also consider fine-tuning the number of Gaussian components again considering the increase in dimensionality due to the appending of delta and acceleration coefficients.

B.5.2 Texture windows

The previous experiment shows that adding some short-term account of the MFCC statistics (i.e. delta or acceleration coefficients) has a positive (although limited) influence on the R-precision. In this paragraph, we investigate the modelling of the long-term statistics of the feature vectors.

It has been shown that, for modeling music, using a larger scale texture window and computing the means and variances of the features over that window results in significant improvements in classification. Tzanetakis in Tzanetakis and Cook (2002) reports a convincing 15% precision improvement on a genre classification task when using accumulations of up to about 40 frames (1 second). This technique has the advantage of capturing the long-term nature of sound textures, while still assuring that the features be computed on small stationary windows (as proved necessary in section B.1.4)

We report here the evaluation results using such texture windows, for a texture window size w_t growing from 0 to 100 frames by steps of 10. $w_t = 0$ corresponds to using directly the MFCCs without any averaging, like in section B.1. For $w_t \geq 0$, we compute the mean and average of the MFCCs on running texture windows overlapping by $w_t - 1$ frames. For an initial signal of n frames of N coefficients each, this results in $n - w_t + 1$ frames of $2N$ coefficients : N means and N variances. We then model the resulting feature set with a M-component GMM. For the experiment, we use the best parameters obtained from section B.1, i.e. $N = 20$ and $M = 50$. Figure B.5 shows the influence of w_t on the R-precision. It appears that using texture windows has no significant influence on the R-precision of our similarity task, contrary to the classification task reported by Tzanetakis : the maximum increase of R-precision is 0.4% for $w_t = 20$, and the maximum loss is 0.4% for $w_t = 10$.

Several directions could be further explored to try to adapt Tzanetakis' suggestion of texture windows. First, the computation of N-dimensional means and variances doubles the dimension of the feature space, hence the optimal number of GMM components M should be adapted accordingly. Second, the use of one single mean (and variance) vector for each window may create a "smearing" of very dissimilar frames into a non-meaningful average. It is likely that using a small size GMM for each texture window would increase the precision of the modelling. However, this raises a number

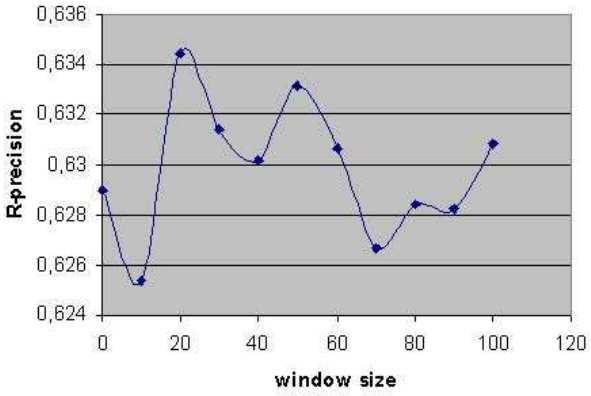

Figure B.5: Influence of the texture window size

of additional issues which were not studied here, among which :

- Which is the optimal number of gaussians, for each frame, and then for the global model ?
- Should the gaussian centres be tracked between neighboring frames ?

Finally, in the single-component case, the mean of the frame-based means (with no overlap) of a signal $\{a_i\}$ is trivially equal to the global mean :

$$\frac{1}{n}\sum_{i=0}^{i=n-1}\frac{1}{m}\sum_{j=im+1}^{j=(i+1)m} a_j = \frac{1}{nm}\sum_{i=0}^{i=nm-1} a_i \tag{B.11}$$

Although the extension of this behaviour in the case of multi-component GMMs cannot be written explicitly (as it results from a learning algorithm), this suggests that the real influence of this processing remains unclear. The extra information captured by texture windows may be more appropriately provided by an explicit segmentation pre-processing, or time-sensitive machine learning techniques like hidden Markov models, as we investigate in section B.5.3.

B.5.3 Dynamic modeling with hidden Markov models

The fact that appending delta and acceleration coefficients to the original MFCCs slightly improves the precision of the measure suggests that the *short-term dynamics* of the data may be an important

factor. Short-term dynamical behavior in timbre may describe e.g. the way steady-state textures follow noisy transient parts. These dynamics are obviously important to compare timbres, as can be shown e.g. by listening to reverted guitar sounds used in some contemporary rock songs which bear no perceptual similarity to normal guitar sounds (same static content, different dynamics). Longer-term dynamics describe how instrumental textures follow each other, and also account for the musical structure of the piece (chorus/ verse, etc.). As can be seen in section B.5.1, taking account of these longer-term dynamics (e.g. by using very large delta coefficients) is detrimental to the similarity measure, as different pieces with same "sound" can be pretty different in terms of musical structure.

To explicitly model this short-term dynamical behavior of the data, we try replacing the GMMs by hidden Markov models (HMMs, see Rabiner (1989)). A HMM is a set of GMMs (also called states) which are linked with a transition matrix which indicates the probability of going from state to another in a Markovian process. During the training of the HMM, done with the Baum-Welsh algorithm, we simultaneously learn the state distributions and the markovian process between states.

To compare HMMs with one another, we adapt the Monte Carlo method used for GMMs : we sample from each model a large number N_S of sequences of size N_F, and compute the log likelihood of each of these sequences given the other models, using equation 3.13. The probabilities $\mathcal{P}(\mathcal{S}_i^{\mathcal{A}}/\mathcal{B})$ are computed by Viterbi decoding.

Previous experiments with HMMs by the authors (Aucouturier and Sandler, 2001a) have shown that models generalize across the songs, and tend to learn short-term transitions rather than long-term structure. This suggests that HMMs may be a good way to add some dynamical modeling to the current algorithm. In figure B.6, we report experiments using a single HMM per song, with a varying number of states. The output distribution of each state is a 4-component GMM (the number of component is fixed). To compare the models, we use $N_S = 200$ and $N_F = 100$.

From figure B.6, we see that HMM modeling performs no better than static GMM modeling. The maximum *R*-precision of 0.632 is obtained for 12 states. Interestingly, the precision achieved with this dynamic model with 4*12=48 gaussian components is comparable to the one obtained with

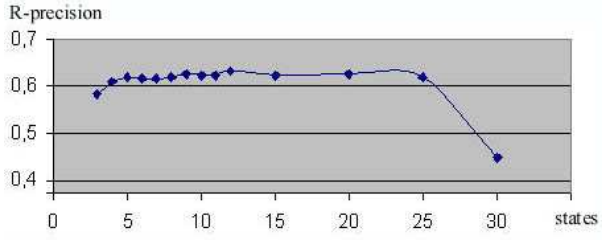

Figure B.6: Influence of the number of states in HMM modelling

a static GMM with 50 states. This suggests that although dynamics are a useful factor to model the timbre of individual monophonic instrument samples (see for instance Herrera-Boyer et al. (2003)), it is not a useful addition to model polyphonic mixtures like the ones we are dealing with here. Probably, the dynamics modeled here by the HMMs are not meaningful, since they are a mix from all the individual sources, which are not synchronised.

B.6 Building in knowledge about note structure (pattern 3.2.8)

We investigate here 2 techniques to build in higher-level knowledge about the structure of musical notes, namely the segmentation between transient and steady state.

B.6.1 Removing noisy frames

Following the intuition of Jiang et al. (2002), we investigate whether removing the percussive and noisy frames in the original signal would improve the MFCC modeling of the music signals. As a pre-processing, we do a first pass on the signal to compute its frame-based Spectral Flatness (Johnston (1988)), with the following formula :

$$SFM_{db} = 10 \log_{10} \frac{G_m}{A_m} \tag{B.12}$$

where G_m is the geometrical mean and A_m the arithmetical mean of the magnitudes of the spectrum on each window. Spectral Flatness is notably used in Speech to segment voiced and unvoiced sig-

nals. Here, we discard frames with a high spectral flatness (using the 3σ criteria) before computing traditional MFCCs on the remaining frames. This crudely amounts to removing the transient part of the notes, and keeping the steady states. This is way to bypass the limitations of MFCCs stressed in Jiang et al. (2002) (poor modeling of the noisy frames), without providing any cure for it, as does e.g. the Spectral Contrast feature variant investigated above.

B.6.2 Note Segmentation

In typical implementations, MFCCs are computed with a constant frame-rate, and thus may average potentially very distinct audio events, namely transient and steady-states of musical notes. In this section, we investigate whether synchronizing the MFCC extraction to the higher-level knowledge of note segmentation can improve the global modelling of polyphonic timbre similarity.

We process each audio file with the custom automatic segmentation algorithm described in Appendix E. We segment each note found by the algorithm into transient and steady-state, by defining the transient as the time it takes to reach 80% of the maximum energy within the note. We then compute one MFCC vector for the transient part, and n_s MFCC vectors for the steady-state, using 2048-point overlapping Hamming windows. This amounts to computing MFCCs with a variable frame-rate, synchronized on the transient parts of the signals. The resulting set of features is then modelled with 50-state GMMs as above.

B.6.3 Comparison of the 2 approaches

Table B.5 shows a comparison of the performance of the two approaches described here with the baseline approach. We observe that both front-end perform only slightly better than baseline, the simple Spectral Flatness filter peaking with an unconvincing 0.7% improvement (absolute). Such improvements are negligible compared to the considerable runtime degradation, notably for the second approach. A possible explanation for these disappointing results is that, even if we carefully segment predominant notes in a polyphonic context -like we do here-, musical background still contributes a number of non-harmonic, transient and non synchronized events that degrade the MFCC

representation. Further investigation into this phenomenon can be found in Chapter 6 (**Experiment 2**).

Table B.5: Influence of Note-structure knowledge

Implementation	*R*-Precision
standard MFCC	0.629
SegMFCC	0.632
SFN	0.636

B.7 Model Equivalence (pattern 3.2.4)

We investigate in this section a number of alternative modelling of the MFCC distribution than the baseline GMM algorithm.

B.7.1 Pampalk's Spectrum histograms

In Pampalk (2004), a simple approach to model the statistics of the spectral shape is proposed and compared to GMM-based models. It is a 2D histogram counting the number of times each loudness level (out of 10 normalized values) is exceeded in a each frequency band (on 20 Bark bands). The histograms are then compared with simple euclidean distance in dim 200. The authors freely distribute their implementation of this method as a Matlab Toolbox*. Although computationally efficient, this method proved significantly worse than the GMM approach in our evaluation framework (*R*-precision: 0.34).

B.7.2 MFCCs Histograms

A common alternative to parametric modelling such as Gaussian models is to use non-parametric algorithms, i.e. which do not require a training stage to estimate optimal values of internal parameters such as gaussian mean and variance. A typical non-parametric model is the simple histogram of the feature data: the data range is divided into a number of bins, and the histograms count the

*http://www.oefai.at/elias/ma

number of feature values in each bin. However, as already illustrated in Section 3.2.1, histograms are extremely subjected to the curse of dimensionality. When the dimension of the feature set increases, the number of bins grows exponentially, which makes the histogram representation of the data extremely sparse. Typical histograms in dimensions greater than 2 or 3 would not have a single non-zero bin in common, which makes them impossible to use for distance computation (and for pretty much anything else). There have been ample research into adaptive sampling methods (Bruno et al., 2001; Thaper et al., 2002; Lim et al., 2003), but state-of-the-art solutions don't manage to raise the conceptual barrier higher than dimension 5. None of such methods are applicable to the 20-dimension space of MFCCs. Therefore, we investigate 3 approaches for adapting the MFCC dataset to the requirements of the histogram method:

- Independent histograms for each dimension: We compute 20 histograms corresponding to each dimension of the MFCC space. The histogram bins are uniformly distributed between the 10% and 90% percentile value in each dimension (absolute min and max data points usually results in spurious values that gives a false estimate of the true data range). Histograms of corresponding dimensions are compared to one another with simple euclidean distance, and the set of distances is averaged to give a final distance measure.

- Vector quantization using GLA: The above method makes the assumption that all MFCC dimensions are independent, and that the co-occurrence of a tuple of values in each dimension at given time steps is uninformative. The discrete cosine transform used in the MFCC algorithm chain is a practical approximation of the *Karhunen-Loeve* transform (Logan, 2000) which tends to decorrelate the different dimensions, however this naturally does not imply that they are independent. In this method, we propose to use vector quantization (VQ) to reduce the MFCC dataset to a 1-dimension, meaningful codebook, which still preserves its multidimensional distribution. The Global-Lloyd algorithm (GLA) is an unsupervised KMean approach that aims at best representing the global distribution of points. The distribution of 20-dim MFCC vectors is clustered by the KMean algorithm (Bishop, 1995), and quantized using the

cluster centres as codebook vectors: each point in the original distribution is quantized to the index of the nearest cluster. The resulting 1-dimension signal is therefore a non-uniform quantization of the original, whose resolution is adapted to the original distribution's density. A 1-dim histogram is then built on the quantized signal using a number of bins equal to the number of codebook vectors. In order to compare the histograms of different songs, a common codebook must be computed for the whole database, using a global set of feature vectors that should be as representative as possible of the total feature distribution.

- Vector quantization using LVQ: We investigate the use of a second type of vector quantization algorithm to reduce the MFCC dataset to a 1-dimension, meaningful codebook for which we can compute histograms: Learning Vector Quantization (LVQ, see Kohonen (1995)). LVQ is a supervised method, related to Kohonen maps, which choose codebook vectors that best account for the boundaries between pre-defined classes. A given number of codebook vectors is initialised for each class. Then a learning phase cycles iteratively through all data points, and updates the closest codebook vector m_c^t to the current instance x^t using the rule:

$$m_c^{t+1} = m_c^t + \alpha(x^t - m_c^t) \tag{B.13}$$

if x^t and m_c^t belong to the same class, and

$$m_c^{t+1} = m_c^t - \alpha(x^t - m_c^t) \tag{B.14}$$

if x^t and m_c^t belong to different classes (with $0 < \alpha < 1$ a learning rate, which may be constant or decrease monotonically with time). If we define the following classification method: *an instance x is decided to belong to class to which the nearest codebook vector m_c belongs*, then it can be shown that the values for the m_i that minimize the misclassification error in the above method are the asymptotic values of the m_i found by the above learning rule. Therefore, LVQ does not attempt to find a codebook that well represents the *distribution* of each class (like the

GLA algorithm), but rather that samples the *boundaries* of each class's Voronoi region. Like for the GLA approach, once the sequence of each song's MFCC has been quantized using LVQ codebook vectors, a 1-dim histogram can be computed and compared to the histograms of other songs. As above, a common codebook must be computed for the whole test database, so the histograms of different songs can be compared.

Optimization of LVQ settings

We have found that the performance of Vector-Quantization approaches, and notably of LVQ, crucially depends on a number of parameters:

- Codebook size: i.e. number of clusters in GLA quantization, and number of vectors in LVQ learning
- Training data size: i.e. the number of MFCCs vectors used to represent the global distribution of features for all possible songs. Both KMean clustering and iterative LVQ learning are too computationally and memory intensive to learn the distribution of all MFCC vectors for all songs (which would amount in the case of our test database to about $3,000,000$ feature vectors). Hence, this total distribution must be downsampled.
- Training data construction: Both GLA and LVQ need to be trained on a representative subset of the total MFCC distribution, i.e. a set of MFCC vectors that encompasses very many different timbres from very many different types of songs. We propose to use the cluster structure of the test database (see 4.2.2) to ensure a high diversity of songs: each song cluster in the database should contribute to the VQ training database. We parameterize the construction of the training data by 2 factors : the number of songs per cluster (N_s) and the number of frames per song which should be kept (N_f). The total training database size is therefore $N_s N_f$ per cluster (with 37 clusters).

The sizes of the optimal codebook and training database are not independent from one another. More codebook vectors are typically needed to model a larger set of feature vectors. Figure B.7

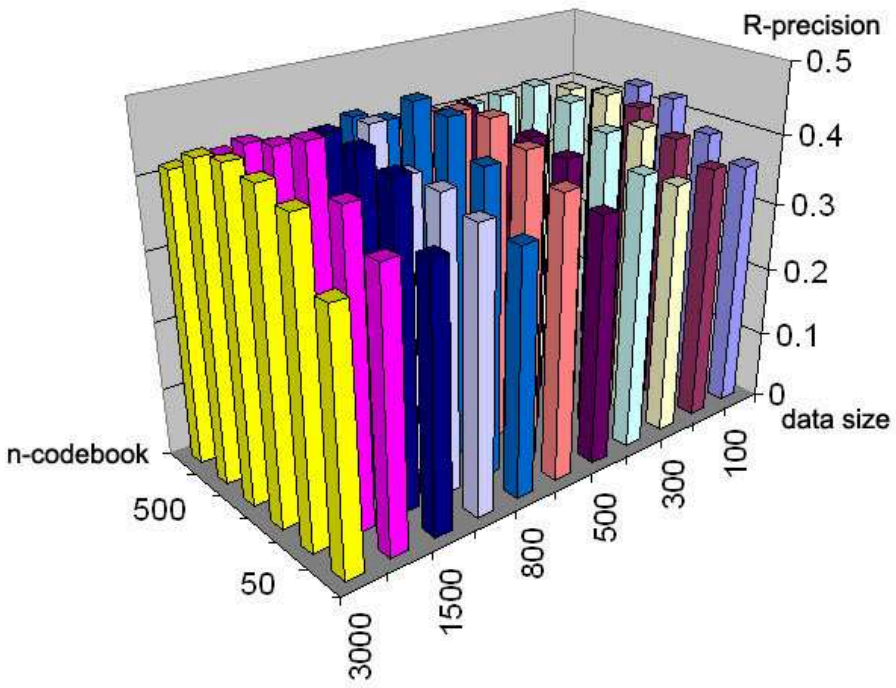

Figure B.7: Precision of Histogram comparison using Learning Vector Quantization for varying number of codebook vectors and training data size (per cluster) (using 100,000 iterations)

shows an exhaustive search over the space of both parameters, measuring the precision of LVQ-based histogram comparison, with a constant number of LVQ training iterations (100,000). It appears that to a certain extent, when the db size increases(from 100 to 800 frames per cluster), the optimal number of codebook vector also increases (from 50 to 500). However, for db sizes above 1000, optimal values are found for 200 codebook vectors. This probably implies that the number of training iterations becomes too small to sufficiently optimize larger codebooks. The best performance (48,8% *R*-precision) is obtained for 200 codebook vectors and 2000 training frames per cluster.

Table B.6 shows the influence of the two parameters N_s and N_f ruling the construction of the training dataset. It appears that the crucial factor is the number of different songs from which the training frames are extracted: more precision is gained by incrementing N_s than by keeping it constant and increasing N_f. This suggests that frames extracted from a given song are relatively redundant for the training process, and that variations between different songs (which may not be correlated to perceptual variations, e.g. bit-rate and width, amplitude normalization, etc.) are crucial to document in the training set.

Table B.6: Influence of training data construction for LVQ histogram comparison. The reported *R*-precision values are the maximum precision over all possible codebook sizes.

nsongs	nframes	data size	*R*-Precision
2	200	400	0.444
2	500	1000	0.463
3	200	600	0.464
3	500	1500	0.466
4	200	800	0.479
4	500	2000	0.488

Comparison of the 3 approaches

Table B.7 shows a comparison of the performance of MFCC histogram similarity, using the optimal settings for the 3 approaches. All 3 approaches use 200-bin histograms, which correspond to 200 k-mean clusters in the case of GLA vector quantization and 200 codebook vectors in the case

of the LVQ. It appears that LVQ performs best for histogram comparison, which is no surprise as the codebook vectors are constructed to best discriminate the different possible sounds. GLA histograms perform significantly worse, which can be explained by the fact that the majority kMean clusters may span dense, common-core areas of the feature space, and thus devote little resolution to the modelling of more informative feature values. Surprisingly, the simple independent approach has a similar performance to the more complex GLA approach: this further suggests that the modelling power of the GLA algorithm is spent on non-informative data, and thus takes little advantage of the statistical dependencies between MFCC dimensions. On the whole, the MFCC histogram approach, while computationally less expensive than Monte-Carlo distance, remains around 15% less precise than the GMM approach with optimal settings.

Table B.7: Comparison of MFCC histogram comparison using 200 bins, and best settings for vector quantization methods.

method	*R*-Precision
Indt Hist	0.41
GLA Hist	0.41
LVQ Hist	0.50

B.8 Borrowing from Image Texture Analysis (pattern 3.2.6)

In this section, we investigate a number of techniques inspired from those used for the automatic analysis of image *textures*.

B.8.1 Image Texture Features

Texture shares this property with musical timbre that it seems to resist simple and consensual definition. It is the characteristic property of images such as found in Figure B.8 which makes them recognizable as e.g. “wood” or “fabric”, and which lies in repeating patterns of the spatial variations of the pixel intensities that translates various physical materials, tactile or light reflection qualities. The modelling of image textures is a much researched area in computer vision, and has

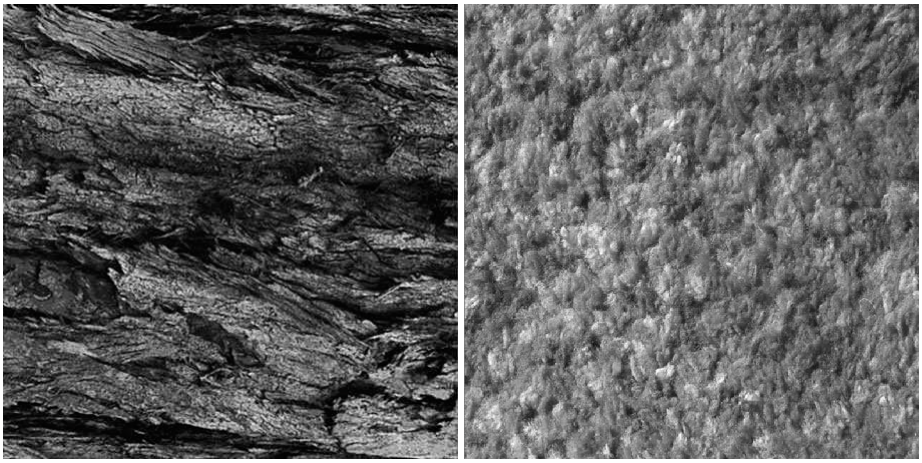

Figure B.8: Two image textures reproduced from MIT VisTex texture repository (MIT, 1995): wood (left) and fabric (right)

multiple applications for classification, retrieval and synthesis of images. For recent reviews of texture research, see e.g. Tuceryan and Jain (1998); Matsumoto and Nishimura (1998). Similarly to monophonic timbre recognition research (as reviewed in Chapter 2), much of the work in texture analysis is grounded on classic psychophysical experiments, most notably the work of Julesz et al. (1973); Julesz (1973). Asking the question "when is a pair of texture images discriminable, given that they have the same brightness, contrast and color", Julesz conjecturates the texture images are not pre-attentively (i.e. effortlessly) discriminable if their second-order statistics are identical. If two textures do not differ by their first and second order moments, but only by their third-order statistics (or higher), their discrimination requires a deliberate cognitive effort. Although complex counter-examples have later been exhibited (Julesz, 1981), second-order moments seem to remain a very salient feature for texture perception, and is therefore exploited by numerous proposals for representing features computationally .

The most well-known and successful feature to exploit the second-order statistics of textures is probably the gray-level co-occurrence matrix (GLCM) as proposed by Haralick et al. (1973). The GLCM of a given image estimates the joint probability $p_{\overrightarrow{d}}(i, j)$ that 2 pixels separated by a *displacement vector* $\overrightarrow{d}$ (i.e. at a distance d along a direction θ) have the gray-level intensity value

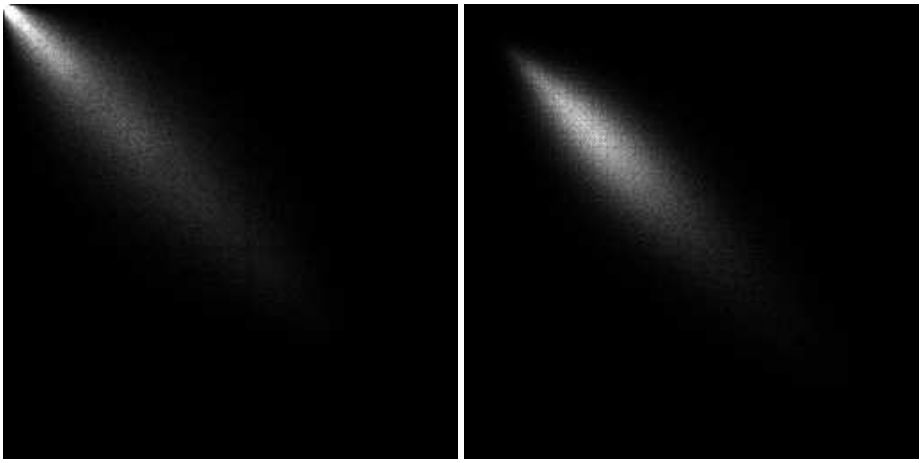

Figure B.9: Grey-level co-occurrence matrices corresponding to the two previous image textures: wood (left) and fabric (right).

i and j. It is computed by quantizing the gray-level values of the image into a number G of bins. More precisely, the GLCM $h_{d,\theta}$ is a G^2 matrix, in which the (i, j) cell is computed by counting the number of duplets of pixels $\{(x, y), (x', y')\}$ separated by $\overrightarrow{d}$ and having their values $f(x, y)$ and $f(x', y')$ equal to i and j respectively:

$$h_{d,\theta}(i, j) = card\{f(x, y) = i, f(x', y') = j\} \tag{B.15}$$

$$where \quad (x', y') = (x, y) + (d\cos\theta, d\sin\theta) \tag{B.16}$$

As GLCM cannot possibly be computed for all values of d and θ, we usually restrain to $d \in 1, 2$ (in pixels) and $\theta \in 0, 45, 90, 135$ (in degrees). Figure B.9 shows the GLCM corresponding to the two image textures in Figure B.8, for $d = 2$ and $\theta = 45$. The matrices exhibit different shapes and distributions along the main diagonal, thus revealing different spatial repetition patterns of the pixels' grey-level values. If most of the entries in the co-occurrence matrix are concentrated along the diagonal, the texture is coarse with respect to the displacement vector $\overrightarrow{d}$.

As G^2-sized GLCMs are too high-dimensional to be compared directly (for computational reasons, but also because they exhibit too much variability), Haralick et al. (1973) have introduced a large number of 1-dimensional secondary features computed from the GLCM, among which we cite

here the 6 most common:

$$Energy \qquad \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} h(i,j) \tag{B.17}$$

$$Contrast \qquad \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} (i-j)^2 h(i,j) \tag{B.18}$$

$$Homogeneity \qquad \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} \frac{h(i,j)}{1+|i-j|} \tag{B.19}$$

$$InverseDifferenceMoment \qquad \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} \frac{h(i,j)^2}{1+(i-j)^2} \tag{B.20}$$

$$Entropy \qquad -\sum_{i=0}^{G-1} \sum_{j=0}^{G-1} h(i,j) \log(h(i,j)) \tag{B.21}$$

$$Correlation \qquad \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} \frac{ijh(i,j)-\mu_x\mu_y}{\sigma_x\sigma_y} \tag{B.22}$$

where μ_x, μ_y and σ_x,σ_y denote the mean and standard deviations of the row and column sums of the matrix, respectively (i.e. of the marginal distributions $p_x(i)$ and $p_y(j)$). GLCM features are typically concatenated into a low-dimension feature vector, and feature vectors of different images are compared to one another using standard euclidean distance. As an example, Lerski et al. (1993) describes a successful application of GLCM features in the context of biomedical image recognition.

B.8.2 Application to Audio

We propose here to apply the same techniques to model the statistics of polyphonic audio textures. Audio signals can be considered as one-dimensional images, each frame of which is equivalent to a pixel in the previous approach. The displacement vector $\overrightarrow{d}$ thus reduces to a temporal lag d between co-occurring frames. We investigate several representation for mimicking the "gray-level" value of an audio frame. As for first-order histograms described in B.7.2 (and even more crudely), co-occurrence matrices are practically impossible to compute for dimensions higher than 1. Hence, only scalar reductions of each frame should be considered.

Individual features

Natural candidates for scalar frame representation are the frame energy (RMS), MPEG7 spectral moment features such as Spectral Centroid, or individual MFCC (1st order, 2nd order). For a

given scalar feature, we compute the co-occurrence matrix $h(i, j)$ by counting the number of audio frames f_t, f_{t+d} separated by d time steps which have, for the given feature, the quantized value i and j respectively. Figure B.10 shows the co-occurrence matrices computed on the frames' Spectral Centroid for a number of polyphonic textures, organized by duplets of similar songs (2 Beethoven piano sonata, 2 acoustic guitar folk pieces and 2 heavy rock tunes). We observe notable similarities in the structure of the matrices for similar songs, and notable differences between songs of different kind. Table B.8 shows a comparison of a number of features on which GLCM can be built. For this experiment, individual features are quantized using 32 bins, compared using a time-step of 1 frame. For each GLCM, a vector composed of the 6 Haralick features described in Equations B.17 to B.22 is extracted, and compared to similarly extracted vectors of other songs using normalized euclidean distance.

The precision achieved by individual features is typically around 15%, which is comparable to the individual performance of similar features for e.g. instrument classification tasks (Eronen and Klapuri, 2000). One can observe that some individual features seem to be better suited to second-order analysis than others: Spectral Centroid for instance is a better basis than RMS for computing frame co-occurrence. Individual features can be combined by concatenating the 6 Haralick features extracted from each GLCM into a $6n$ feature vector. One can see that combining MP7 spectral moments improves the precision. This is even clearer when combining the features of each of the first 20 MFCCs, which gives a precision of 36%. However, further combining spectral moments-based and MFCC-based Haralick features doesn not further improve the precision.

Influence of size, time step and distance algorithm

We investigate here the influence of various parameters on the performance of the associated distance measure :

- Time steps: The lag in frame number used to compute co-occurrence scores. Table B.9 shows that time steps from 1 to 5 have little average influence on the measure.

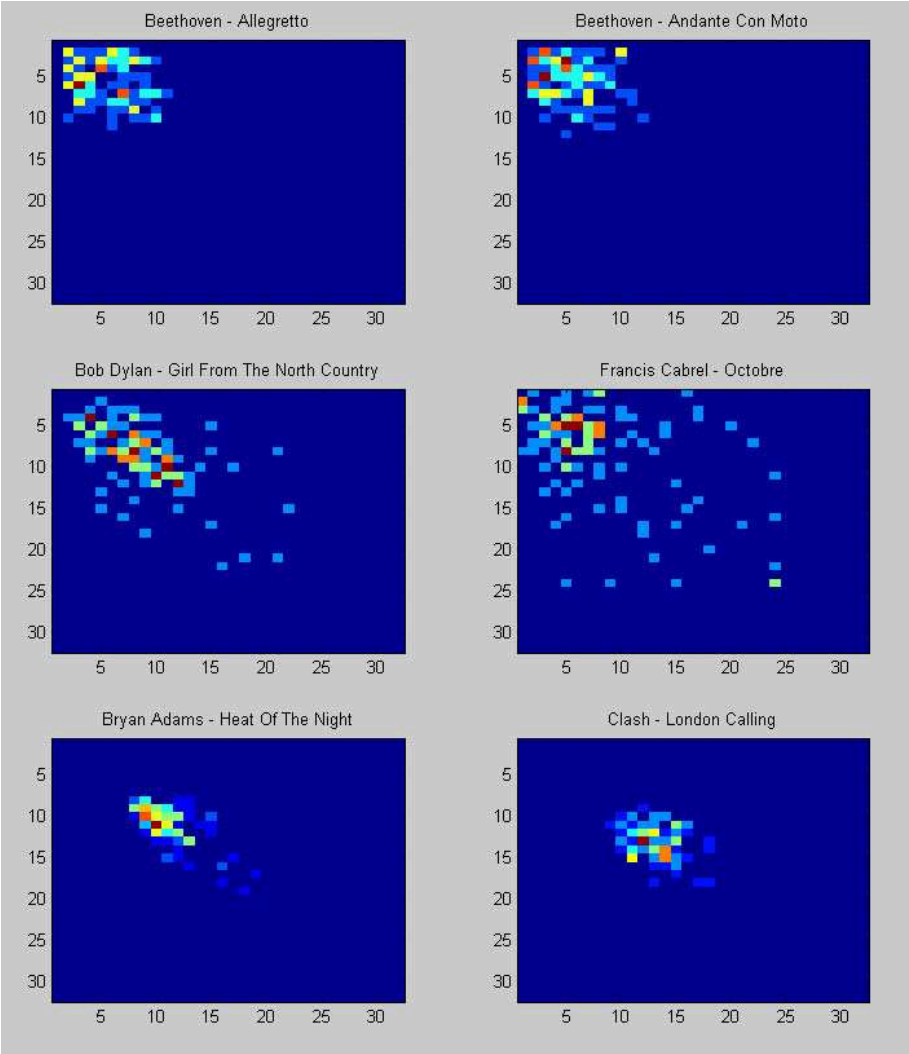

Figure B.10: Spectral Centroid Co-occurrence matrix for a number of polyphonic textures.

Table B.8: Comparison of features for GLCM feature comparison.

feature	*R*-Precision
RMS	0.11
Spectral Rolloff	0.13
Spectral Flatness	0.14
Spectral Spread	0.15
Spectral Centroid	0.17
SC+SR+SF	0.20
1-order MFCC	0.14
2-order MFCC	0.13
Combined 20 MFCC	0.36
20 MFCC+SC+SR+SF	0.30

Table B.9: Influence of the number of time steps on the *R*-precision of GLCM comparison based on the Spectral Flatness descriptor.

Feature	1-frame step	2-frame step	5-frame step
Spectral Flatness	0.140	0.139	0.140

- Number of Bins: The number of discrete values used to quantize the feature value representing each frame, and from which co-occurrences are found. Table B.10 shows that resolutions from 5 to 64 have little influence on the measure.

Table B.10: Influence of the number of histogram bins on the *R*-precision of GLCM comparison based on the Spectral Flatness descriptor.

Feature	5 bins	32 bins	64 bins
Spectral Flatness	0.135	0.14	0.138

- Distance Measure: GLCM feature vectors, using the 6 Haralick features described in Equations B.17 to B.22 are compared with euclidean distance. We test here 3 implementations of the distance:

  – *absolute* (A), where feature vectors are compared without any normalization:

$$d(a,b) = \sum_i (a[i] - b[i])^2 \tag{B.23}$$

– *normalized* (N), where feature vectors are first normalized between 0 and 1 in each dimension, using minima and maxima computed on the whole testing database:

$$d(a,b) = \sum_i (a_N[i] - b_N[i])^2 \tag{B.24}$$

$$a_N[i] = \frac{a[i] - min[i]}{max[i]} \tag{B.25}$$

$$b_N[i] = \frac{b[i] - min[i]}{max[i]} \tag{B.26}$$

– *relative* (R), where feature vectors are compared with increase ratios (thus relaxing the need for global bound computation):

$$d = \sum_i \frac{(a[i] - b[i])^2}{min(a[i], b[i])} \tag{B.27}$$

Table B.11 shows that normalized euclidean distance performs best, but that relative distance can be used when global bound computation is not practical.

Table B.11: Influence of the euclidean implementation on the *R*-precision of GLCM comparison based on the Spectral Flatness descriptor.

Features	A	N	R
Spectral Flatness	0.08	0.14	0.12

B.8.3 Vector Quantization

As seen in Section B.8.2, combining Haralick features for GLCMs computed on each dimension of a 20-dim MFCC set leads to a large improvement over individual features considered alone. However, this method makes the assumption that all MFCC dimensions are independent, and that the co-occurrence of a tuple of values in each dimension at given time steps is uninformative. Therefore, like for 1st-order histograms in Section B.7.2, we propose to use vector quantization (VQ) to reduce the MFCC dataset to a 1-dimension, meaningful codebook, which still preserves its multi-dimensional distribution. A unique GLCM can then be computed on the quantized feature signal,

corresponding to a unique set of 6 Haralick features, which can be compared to neighbor vectors using euclidean distance.

We compare here the 2 VQ methods (GLA and LVQ) introduced in Section B.7.2, with their optimal settings found above. Table B.12 compares the precision of both methods, in various settings.

Table B.12: Comparison of features for GLCM feature comparison.

Algorithm	*R*-Precision
LVQ GLCM Features	0.06
GLA GLCM Features	0.07
LVQ GLCM direct	0.44
GLA GLCM direct	0.48
LVQ Histogram	0.50
GLA Histogram	0.41
Combined 20 MFCC GLCM Features	0.36

We observe the following facts :

- Haralick features do not work: The classic Haralick features described in Equations B.17 to B.22 are extremely detrimental to the precision of the measure, when used on quantized data (7% *R*-precision).

- VQ better than individual CM: We therefore also investigate the direct Euclidean comparison of the co-occurrence matrices, without reducing them using Haralick features. The performance achieved by this method (48%) is more than 10% greater than that obtained in Section B.8.2 by combining individual dimensions.

- GLA better than LVQ for co-occurrences: Direct CM comparison shows that GLA-based matrices are better representations than LVQ-based matrices.

- Co-occurrences no better than 1st-order: Table B.12 shows a comparison of the performance of MFCC-based CMs with simple first-order histogram similarity using both vector quantization methods. It appears that the best CM distance (using GLA) is no better than the best dis-

tance based on simple 1st-order histograms (using LVQ) (48% against 50%). It also appears that, contrary to CM comparison, LVQ performs better than GLA for histogram comparison.

- No improvement over GMMS of MFCCs: On the whole, the Vector Quantization of MFCCs and their modelling, either static with first order histograms or dynamical using Image Texture features, while computationally less expensive than GMM modelling, remains around 15% less precise than the GMM approach with optimal settings, as reported in Section B.1.

B.8.4 Conclusions of texture analysis

Haralick features not meant for Vector Quantization

Haralick features are successful representations for gray-level co-occurrence matrices, because they mainly describe the distribution spread along the diagonal. However, they are poor features for co-occurrence matrices based on vector quantized data. Contrary to uniform quantization in one dimension, the quantized data resulting from vector-quantization of multidimensional features is non-ordered: neighboring bin values have no relation to one another. Therefore, no particular distribution can be expected along the main diagonal of the co-occurrence matrix (expect of course the first diagonal itself, corresponding to self co-occurrence).

Vector Quantization successfully captures multi-dimensional co-occurrences

The fact that the performance of VQ-based CMs is greater than the performance obtained by combining the CMs of individual dimensions shows that Vector Quantization is a meaningful representation for computing multidimensional co-occurrences.

LVQ not optimal for co-occurrence analysis

However, the specific LVQ-type of vector quantization does not seem to provide a good representation for computing co-occurrences. LVQ-codebook vectors are optimized to best discriminate the different possible sounds, and thus exhibit little overlap from texture to texture. GLA-codebook

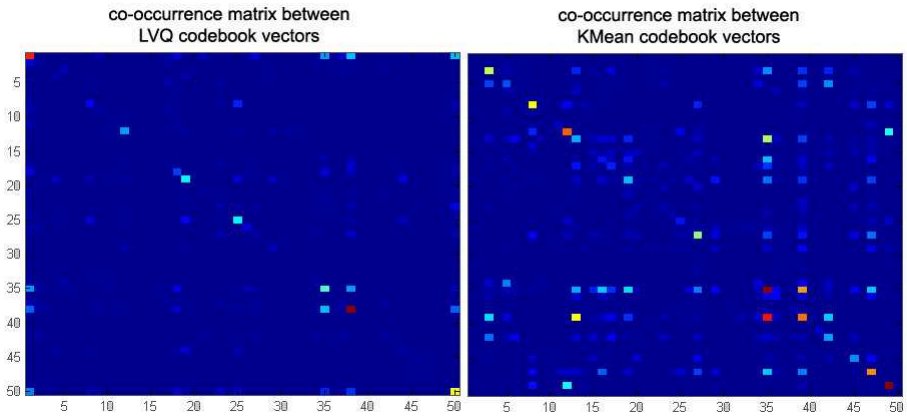

Figure B.11: Co-occurrence matrices based on the same set of 20-dim MFCCs using 2 vector quantization methods: LVQ (left) and GLA (right). The LVQ-based matrix is sparser than the GLA-based one, thus yielding a poor representation for co-occurrences.

vectors however are unsupervised k-means cluster centers, which tend to spread over the whole distribution of MFCC points, thus ensuring that the quantized representation overlaps between textures, and that co-occurrence matrices can be compared with greater precision. Figure B.11 shows two co-occurrence matrices based on the same set of 20-D MFCCs using the two vector quantization methods. It illustrates the critical sparsity of LVQ-based CMs, and the more homogeneous density of GLA-based CMs.

But LVQ is optimal for first-order histograms

Contrary to CM comparison, LVQ performs better than GLA for histogram comparison. The poor overlap of LVQ basis creates too much sparsity in 2-dimensional co-occurrence analysis, but is beneficial for 1-dimensional histogram comparison as the codebook vectors are constructed to best discriminate the different possible sounds. GLA histograms perform significantly worse, which can be explained by the fact that the majority of k-means clusters may span dense, common-core areas of the feature space, and thus devote little resolution to the modelling of more informative feature values.

Co-occurrence analysis is no better than 1st-order

The best CM distance (using GLA) is no better than the best distance based on simple 1st-order histograms (using LVQ) (48% against 50%). This suggests that 2nd-order statistics as analysed by CM, are not a factor as crucial for the comparison of sound textures as it is for image textures. This is reminiscent of the identical performance achieved by GMMs and HMMs with the same number of Gaussian components. This indicates that co-occurrence analysis fails to account for any meaningful dynamics in audio data, and is at best equivalent to a static model with similar degrees of freedom.

Comparison with the "GMMS of MFCCs" approach

The Vector Quantization of MFCCs and their non-parametric modelling, either static with first order histograms or dynamical using image texture features, is around 15% less precise than the parametric approach with GMMs. However, the former is a lot less computationally demanding than the latter. The computation of distances between GMMs is based an expensive Monte-Carlo estimation of the Kullback Leibler divergence, which is unrealistic to compute on the fly. Hence, GMM distances between songs need to be pre-computed and stored in large matrices in order to be used in query systems. Although we have proposed efficient ways to do so in Roy et al. (2005), this practically rules out the scaling up of such systems to very large music databases (e.g. of several 100,000 music titles). VQ-based histograms and CMs are compared using much simpler Euclidean distances, which are easy to optimize using multidimensional index structures such as KD-trees, and thus can be computed on the fly for very large database without having to store any precomputed distance matrix.

Appendix C

Comparison of implementation performance

While many authors, such as Pampalk (2004), rely on Matlab implementations of the various algorithms, it appeared in this study that runtime performances were critical in order to enable the testing of many algorithm parameters over large ranges of values. Moreover, the need for large database architecture and metadata-management tools posed the additional problem of interoperability between the algorithm implementations and the Java-based tools such as MCM and the MB. Finally, an additional constraint is the flexibility to modify the implementations in order to test variants, which tends to favour proprietary implementations compared to third-party toolboxes.

Therefore, we investigated a number of alternative and faster implementations, both for feature extraction, distribution modelling and distance computations, which we compare here.

C.1 Feature Extraction

Several implementations were developed and tested for feature extraction, namely “typically” MFCC computation.

- MATLAB custom code: For the first prototype described in Aucouturier and Pachet (2002b), we developed our own matlab code for MFCC extraction, of which we give here a schematized view:

```
z=windowing(s,n,inc); % n-size hamming windows
f=rfft(z.'); % real-part fft
[m,a,b]=melbank(nchan,n,fs); % mel filterbank
pw=f(a:b,:).*conj(f(a:b,:)); % power spectrum
y=log(m*pw); % log of filtered spectrum
c=rdct(y).'; % discrete cosine transform
```

- MATLAB slaney: We tested an alternative matlab implementation of the MFCC algorithm, provided my Malcolm Slaney's Auditory Toolbox (Slaney, 1998). More than our custom code, this implementation relies on Matlab vectorized array manipulation routines, which supposingly makes it more efficient, but also more difficult to read. An illustrative example of this rationale is the way the Mel filter bank coefficients are computed:

```
mfccFilterWeights(chan,:)  = ...
(fftFreqs > lower(chan) & fftFreqs <= center(chan)).* ...
triangleHeight(chan).*(fftFreqs-lower(chan))/(center(chan)-lower(chan)) + ...
(fftFreqs > center(chan) & fftFreqs < upper(chan)).* ...
triangleHeight(chan).*(upper(chan)-fftFreqs)/(upper(chan)-center(chan));
```

The expression `(fftFreqs > lower(chan) & fftFreqs <= center(chan))` designs a vector constructed by a vectorized logical AND between 2 vectors `a = (fftFreqs > lower(chan))` and `b = (fftFreqs <= center(chan))` which are themselves constructed by vectorized logical comparison operators. `a` is a vector of 0 and 1, of the same size as `fftFreqs` (which lists all

sampled frequencies in the spectrum), and containing 0 for all frequencies below `lower(chan)` and 1 above. `a` is intersected with `b` which similarly contains 0 for all frequencies above `center(chan)` and 1 below. This results in a vector containing 1 between `lower(chan)` and `center(chan)` and 0 elsewhere. Thanks to vectorized statements, this vector is constructed in 2 comparison operations, and one multiplication, while a naive implementation would have required $2n$ comparisons, where n is the size of `fftFreqs`. This binary vector is then multiplied by a linear increasing function of the frequency, and summed piece-wise with a symmetric descending linear function, which finally results in a triangular-shaped filter between `lower(chan)` and `upper(chan)`.

- Compiled MATLAB: The previous Matlab implementations cannot be easily integrated into the Java framework described in Section 4.3. However, Matlab provides the option to automatically generate C/C++ code from matlab scripts. The code dynamically links to a large set of proprietary dlls implementing the various libraries available in the Matlab environment. From the machine-generated code (which is typically difficult to read), we then can generate e.g a Windows executable file that can be called from Java through the OS, by invoking the `java.lang.Runtime` class's `exec()` method. Note that, due to the limited buffer size for standard input and output streams, the above method has to be combined with a thread that constantly reads and dumps the input, output and error streams of the subprocess for it not to block, and even deadlock (Monk et al., 2000):

```
Runtime rt = Runtime.getRuntime(); // get Runtime object depending on native environment
Process proc = rt.exec(myCmdString);// execute the native command, e.g. ''mfcc.exe''
Thread reader = new Thread(){// constantly reads out the process's output stream
   public run(){
      InputStreamReader isr = new InputStreamReader(proc.getInputStream());
      BufferedReader br = new BufferedReader(isr);
      String line = null;
      while ((line = br.readLine()) != null){}
   }
};
reader.start();
```

- HTK : The Hidden Markov Model Toolkit (HTK, Young et al. (1993)) is a toolkit designed for speech recognition research, which allows building and manipulating hidden Markov models. It consists of a set of library modules and tools available in C source form. The library includes a number of speech feature extractors, among which a fast MFCC implementation. As with compiled matlab code, HTK can be used to compile a Windows executable being called from the OS via the Java Runtime API. Although the source code of HTK is available, re-using and modifying the MFCC-part of the code outside of the the whole HTK framework is difficult, partly because it is integrated in a `HCopy` module which unifies all feature extraction processes, and uniquely works on the HTK file format. Hence, this is a case of a fast implementation that is nevertheless limited in its flexibility in our context.

- Libedso : `libedso` (standing for "Library of EDS operators") is a set of library modules written in C, and developed in Sony CSL for the EDS project (Pachet and Zils, 2003). It includes many feature extractors, including MFCCs, and reifies a number of data structures such as `edso_matrix`, meant to simplify the portability from Matlab implementations. As for the previous implementations, `libedso` can be used to compile a standalone executable which can be called from the OS through the Java Runtime API. However, due to the perfect control on source code and the implementation modularity, it is also possible to call native `libedso` code directly from Java, with *Java Native Interface* (JNI), which is supposingly more robust and also enables more advances I/O operations that `Runtime.exec()`. Both interfacing options were tested.

Table C.1 reports the runtime in seconds of the various implementations for MFCC extraction. All test were conducted on the same audio file, *The Beatles - A Hard Day's Night* (2min32, stereo, 44100 Hz). 20 MFCCs were extracted from 32 Mel bands, using 50% overlapping 2048-point hamming windows. One can see that Matlab code performs 4-5 times slower than native implementations. A small advantage is gained from vectorized operations in the optimized Slaney script over

the custom script, however the benefits of vectorization are lost when using the Matlab compiler. The compiled version of Slaney's script actually runs slower than the original Matlab version, which is probably due to a poor automatic compilation of the vectorized instructions, which above a certain level of handled complexity, are probably simply unwrapped to a series `for` loops. The native implementations of HTK and libedso have similar run-time when called with the `java.lang.Runtime` API. Interfacing `libedso` with JNI seems to be associated with a overhead, which makes it slower than the `java.lang.Runtime` versions. However, JNI code has the advantage of greater flexibility: algorithm variants can be written direclty in Java without the need to write and compile new executables.

Table C.1: Comparison of runtime (in seconds) for various implementations of MFCC extraction (see main text for parameters).

Implementation	**cpu-time**
Matlab (custom)	22.48
Matlab (slaney)	22.01
Compiled Matlab (custom)	18.23
Compiled Matlab (slaney)	23.98
HTK	3.85
libedso (native)	3.87
libedso (JNI)	5.73

C.2 Distribution Modelling

Similarly, we tested various implementations for the distribution modelling stage of the similarity algorithm, notably Gaussian Mixture Model training with the EM algorithm. These implementations are also used for the distance computations between models using Monte-Carlo sampling.

- Matlab: The prototype of Aucouturier and Pachet (2002b) relies on a third-party Matlab toolbox for pattern recognition, *Netlab* (Nabney, 2001). This implements standard E-M training, as well as K-Mean initialization for GMM models with either diagonal or complete covari-

ance matrices.

- Compiled Matlab: As proposed above for MFCC extraction, the Netlab scripts can be automatically compiled into C code, which can then be built into Windows executables and called with the `java.lang.Runtime` API.

- Torch: *Torch* is an object-oriented machine-learning library written in C++ (Collobert et al., 2002). It provides a native implementation of Diagonal-covariance GMM models.

- Libedsm: `libedsm` (standing for "Library of EDS Models") is the companion library for `libedso`, i.e. a proprietary machine-learning library providing notably plain C implementations of GMM models. The code offers more flexibility for modification than the Torch implementation and is optimized for speed, notably due to its specificity (compared to the complicated and generic O-O architecture of the TORCH library).

Table C.2 compares the runtime (in seconds) of the various GMM implementations, for both GMM training and distance computation. All training tests were conducted using the same sets of 20-dim MFCCs extracted from *The Beatles - A Hard Day's Night* (2min32, stereo, 44100 Hz). GMMs were trained with 10 gaussian components, using 50 K-Mean iterations (initialisation) and 500 E-M iterations (training). Distance tests report the time to compute distances from the resulting GMM to a common set of 100 similar 20-components GMMs, using 2000 samples drawn from each distribution. One can see that Matlab implementations are up to 20 times slower than native implementations, which makes them unrealistic for large-scale testing. There is a large gap between runtimes of the Matlab scripts and their Matlab-compiled versions, compared to the improvement reported in Table C.1 for MFCC extraction. This may be explained by memory management issues: the Matlab environment itself consumes more memory space than the native runtime environment, which leaves less space for the memory-intensive training process. Torch performs slightly faster than our `libedsm` native implementation for the training of individual GMMS, however is surprisingly nearly 3 times slower for distance computations. This may be explained by our inelegant re-use of the Torch architecture for our specific purpose: collecting the likelihoods of each data

frame from a Torch models requires to inspect a number of data structures after the probabilities are first computed, while likelihoods are gathered and summed on the fly in our custom implementation. Moreover, our implementation adds a number of Monte-Carlo optimisations (described in Roy et al. (2005)), which further speeds up the sampling process. The number of GMM training equals the number of songs in the testing database, while the number of distances computations between GMMs is quadratic. Therefore, the improved performance on distance computation, at the expense of slightly less performance for training, is an important factor in favour of the `libedsm` implementation.

Table C.2: Comparison of runtime (in seconds) for various implementations of GMM training and Monte-Carlo distance between GMMs (see main text for parameters).

Implementation	**cpu-time (training)**	**cpu-time (distance)**
Matlab (Netlab)	487.2	18.25
Compiled Matlab (Netlab)	135.4	11.05
TORCH	2.4	5.96
libedsm (native)	3.06	2.16

Appendix D

Nearest Neighbor Algorithm

Even with fast optimized implementations for distance computations, the task of finding the nearest neighbors (NN) of a given song among large sets of songs (typically several ten of thousands) is a very costly operation. This operation is needed both for the exploitation of a given similarity metric ("find me songs that sound like X") and for the repetitive evaluation of algorithmic variants that we propose to do in this study. This performance bottleneck is one of the principal reasons for the lack of systematic evaluation found in the literature.

In this appendix, we describe a generic algorithm for fast NN search in metric spaces*. The algorithm exploits an intrinsic property of the class of similarity algorithms that we study here: all exhibit a *precision-cputime tradeoff* for some parameter p (*tradeoff parameter*), i.e. for which both the precision and the cputime increase with p.

D.1 Tradeoff between Precision and CPU-time

Many candidates exist for the tradeoff parameter p:

- p may be the size of the feature vector. As already described, the number of MFCCs typically influences the precision of the measure, but also the dimension of the model, hence the cpu

*Parts of these results were reported in Roy et al. (2005)

time both for learning and comparing.

- p may also be the size of the model, e.g. the number of gaussian components in a GMM, or the bin size of a histogram. The more complex the model, the more precise the measure[†], but also the more expensive the learning and the comparison.

- p can also be found at the model comparison stage. In the case of Monte-Carlo approximation of the Kullback Leibler distance between GMMs, the more samples are drawn from the GMMs, the more precise is the approximation by virtue of the central limit theorem, but also the more expensive are both the sampling and the distance computation.

Note that many other music-related distance measures, such as melodic similarity, also exhibits such a precision-cputime tradeoff. In the case of dynamic-programming based measures, a possible choice for p is the size of the alphabet used to describe the items to be compared. For instance, melodic comparison often rely on some quantization of the pitch, using either the exact pitch, or more and more approximated intervals, up to a simple {up, down} contour representation. Ito et al. (2004) shows how the quantization error degrades the precision of a Query-by-humming query.

We propose to exploit the precision-cputime tradeoff of such distance algorithms $\mathcal{A}$ to efficiently calculate the result of NN queries. We use n successive refinements of $\mathcal{A}$ to compute first cheap, unprecise distances (i.e. $\mathcal{A}(p)$ for p small) on the whole set of possible items, then more and more expensive and precise distances (i.e. $\mathcal{A}(p)$ for p big) on smaller and smaller sets. If the precision $PREC(p)$ of the distance measure increases *faster*[‡] than the cputime $CPU(p)$, then we will show that the cumulated cpu time of the successive steps using $\mathcal{A}(p_0)$, $\mathcal{A}(p_1)$, ..., $\mathcal{A}(p_{n-1})$ may be a lot smaller than the direct computation of the most precise distance $\mathcal{A}(p_{n-1})$ on the whole set of items.

This approach can be viewed and implemented as a planning wrap-up around an existing distance measure, to speed up the associated nearest neighbor search. We show that dramatic speed-up can be achieved without modifying the implementation of the underlying distance measure.

[†] this is not taking into account the curse of dimensionality, see earlier

[‡] in a sense to be defined in Section D.2

D.2 Algorithm formulation

We are interested in computing the elements of a set $\mathcal{S}$ that satisfy a given criterion c, called the target criterion. Note that computing the NN of a given item with respect to a given measure is a particular instance of this schema. The approach we present applies to any target criterion that can be approximated by a series of criteria with the following property: rough approximations of the target criterion are easy (fast) to compute, whereas good (precise) approximations of the target criterion take longer. Moreover, approximations are faster to compute than the target criterion itself.

The standard approach to computing the set N of elements of $\mathcal{S}$ that satisfy c is to evaluate c against every item in $\mathcal{S}$, retaining only those items that satisfy c. Roughly speaking, our approach consists in starting with a first criterion that can be evaluated quickly, to eliminate irrelevant items, and then, to progressively evaluate criteria that are better approximations of the target criterion, finishing with the target criterion itself, to achieve the task. The idea behind this strategy is that if the precision of the successive criteria increases faster than their computation cost, we can save a substantial amount of computation time, because criteria that are expensive to evaluate will be evaluated against fewer items.

D.2.1 Definitions and Assumptions

Let us first introduce some necessary definitions and conventions:

- $\mathcal{S}$ is a finite set.
- c is a criterion defined over $\mathcal{S}$

$$c : S \rightarrow \{true, false\} \tag{D.1}$$

  We call c the *target* criterion.

- c_0, c_1, ..., c_n are criteria defined over $\mathcal{S}$ that approximate c with increasing precision, with the convention that $c_n = c$.

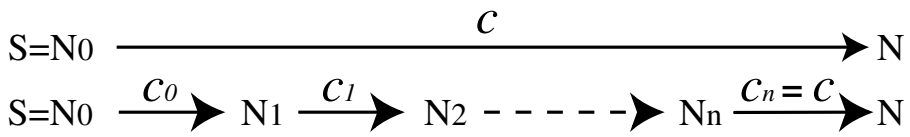

Figure D.1: Instead of computing N directly by applying c, we iteratively compute the N_i for $i = 0, 1, ..., n$

- N is the subset of $\mathcal{S}$ containing those elements that satisfy c. The goal of the algorithm is to compute N.
- Similarly, N_i is the subset of $\mathcal{S}$ that contains those elements that satisfy c_{i-1}. By convention, we define $N_0 = \mathcal{S}$.
- $t(c_i) < t(c_{i+1}) \forall i \in [0, n-1]$ where $t(c_i)$ denotes the cpu time needed to compute $c_i(x)$ for any element $x \in \mathcal{S}$.

Note that the two following properties are a formalization of the class of algorithm presenting a precision-cputime tradeoff:

Property 1 *c_0, c_1, ..., c_n approximate c with increasing precision*

Property 2 *The cost of computing c_i increases with i, i.e. $t(c_{i+1}) > t(c_i)$*

The NN-algorithm can be described by a simple idea, illustrated in Figure D.1: instead of computing N directly by applying c, we iteratively compute the N_i for $i = 0, 1, ..., n$.

Property 3 *$N_n \subseteq N_{n-1} \subseteq ... \subseteq N_1 \subseteq N_0 = \mathcal{S}$ (i.e. $c_{i+1} \Rightarrow c_i$)*

When Property 3 holds on the N_i sets, it is straightforward to show that $c_i(N_i) = c_i(\mathcal{S}) = N_{i+1}$. In other words, one can compute N_{i+1} by applying c_i to N_i instead of applying c_i to $\mathcal{S}$, thus saving time since N_i is smaller than $\mathcal{S}$.

Figure D.2 illustrates the algorithm. In this figure, we assume that $\mathcal{S} = N_0 = \{x_1, x_2, ..., x_p\}$ and that the x_i are ordered so that $N = \{x_1, x_2, ..., x_k\}$ and more generally $N_i = \{x_1, x_2, ..., x_{k_i}\}$. This

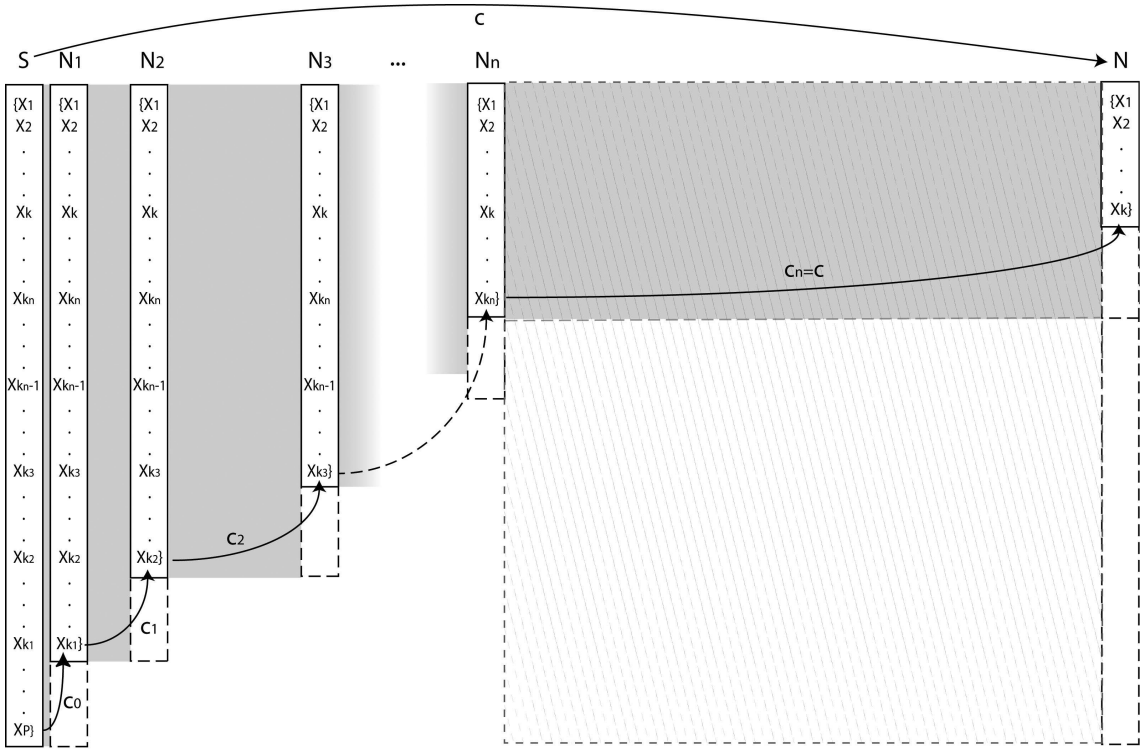

Figure D.2: Illustration of the algorithm. The cumulated cost of the successive steps appears as the light gray area, whereas the cost of the direct NN calculation appears as the stripped area.

reordering is made possible by the inclusion relationship between the N_i sets assumption. The top part, with the horizontal arrow labeled $c = c_n$, represents the standard way of computing N, i.e. evaluate c on every element of $\mathcal{S}$, and retain only the items that satisfy c. The cost of this approach is:

$$t(c)|\mathcal{S}| = t(c)|N_0| \tag{D.2}$$

where $t(c)$ is the time it takes to evaluate function c on one item and $|\mathcal{S}|$ is the cardinality of $\mathcal{S}$. The rest of the figure illustrates our approach, reading from left to right. The leftmost column of the figure, labeled "$\mathcal{S}$" is an enumeration of $\mathcal{S}$. The nearest column, labeled "N_1", can be understood as follows: we evaluate c_0 on every item in $\mathcal{S}$, which yields N_1, the set of items that satisfy c_0. This is represented by the oblique arrow labeled "c_0". N_1 is enumerated vertically in this column. The cost

of this step is:

$$t(c_0)|\mathcal{S}| = t(c_0)|N_0| \tag{D.3}$$

Reading Figure D.2 from left to right illustrates that we iteratively apply c_0, c_1, ..., c_n to N_0,N_1, ..., N_n. Eventually, $c_n = c$, the target criterion, is evaluated against N_n, yielding N. The overall cost of this approach is the sum of the cost of each step:

$$\sum_{i=0}^{n} t(c_i)|N_i| \tag{D.4}$$

Our approach is interesting only in those situations where:

$$\sum_{i=0}^{n} t(c_i)|N_i| < t(c)|N_0| = t(c).|\mathcal{S}| \tag{D.5}$$

On Figure D.2, the successive sets computed are represented vertically, and the successive criterion evaluations are represented horizontally. The costs can be visualized graphically if we assume that the proportions are respected, i.e. that the height of a set is proportional to its cardinality and that the width of a column is proportional to the cost of the corresponding criterion evaluation. The overall cost of our approach corresponds to the light gray surface (the upper-left “triangle”), while the cost of the standard approach is the hashed surface. With this graphical representation, it appears that if the N_i (the heights) decrease fast enough and that the $t(c_i)$ (the widths) simultaneously increase fast enough with increasing i, the light gray surface will be substantially smaller than the hashed surface. This is what we discuss in the next section.

D.2.2 Efficiency

Our approach is interesting when it saves time, i.e. when equation D.5 holds. This gives us a set of necessary conditions for the method to run faster than the standard approach. We will construct

them recursively on n, starting with the case $n = 1$. For $n = 1$, equation D.5 becomes:

$$t(c_0)|N_0| + t(c_1)|N_1| < t(c).|\mathcal{S}| \tag{D.6}$$

$$\Rightarrow \quad t(c_0) < t(c)\frac{|N_0|-|N_1|}{|N_0|} = t(c)\frac{|\mathcal{S}|-|N_1|}{|\mathcal{S}|} \tag{D.7}$$

For $n = 2$, equation D.5 becomes:

$$t(c_0).|N_0| + t(c_1).|N_1| + t(c_2).|N_2| < t(c).|S| \tag{D.8}$$

where $c_2 = c$. If we assume that equation D.7 holds, we get the sufficient condition:

$$t(c_1) < t(c)\frac{|N_1| - |N_2|}{|N_1|} \tag{D.9}$$

and so on. Finally, we have the following sufficient conditions for our approach to be interesting in terms of computation time:

$$t(c_i) < t(c)\frac{|N_i| - |N_{i+1}|}{|N_i|}, \forall i \in [0, n-1] \tag{D.10}$$

$|N_i|$ is related to the precision with which c_i approximates the target criterion c: the less precise is c_i, the larger is the smaller set of items that satisfy c_i which contains all items that satisfy c. Equation D.10 thus requires that at each step i, the precision of the c_i's increases faster than their complexity.

D.2.3 Implementation

For a given problem, one thus needs to find a sequence of steps (the successive c_i's and N_i's) that both verifies properties P_1, P_2, and P_3 and equation D.10. Equation D.10 holds on the cardinalities of the successive result sets (the N_i sets). Therefore, our approach is worth applying to problem for which the cardinalities of the result sets can be computed or estimated easily. This is the case for

the class of similarity measures considered in this work, as will be seen in Section D.3.

For a given set S and a given criterion c, our approach is based on the existence of a series $(c_i)_i$ that satisfies properties P_1, P_2, and P_3. Such a series can easily be found for the class of criteria that possess a tradeoff parameter p. Let us assume that p takes value in a finite set $P = \{0, ..., n\}$ (using quantization if needed). In this context, the series $(c_i)_{i \in P}$ does not necessarily satisfy equation D.10, and if it does, there may exist sub-series of $(c_i)_{i \in P}$ that allow a more efficient implementation of our approach. More precisely, given set S and criterion series $(c_i)_{i \in P}$, there exist 2^n sub-series $(c'_i)_{i \in P' \subseteq P}$ of $(c_i)_{i \in P}$, corresponding to different steps of the approach. (Note that if $(c'_i)_i$ is a sub-series of $(c_i)_i$, an item c'_j is one of the c_i with $j \leq i$, and similarly, $N'_j = N_i$.) The cost of the approach for $(c'_i)_i$ is $\sum_{i \in P'} t(c'_i)|N'_i|$. Among those sub-series, at least one of them is optimal, i.e. there is at least one for which minimizing $\sum_{i \in P'} t(c'_i)|N'_i|$. Note that when the optimal sub-series contains only the target criterion c, our approach equals the standard approach.

To implement the approach optimally, one needs to compute the optimal $(c'_i)_i$. In general, one cannot compute the cost of every 2^n sub-series. However, this can be achieved very efficiently using dynamic programming, as illustrated by the following algorithm:

```
bestSubSeries(n)
   if memValue(n)already computed
      return memValue(n)
   min ← +∞
   for p ← 0 to n − 1
      tmp ← bestSubSeries(p)
      c ← cost(tmp ∪ {n})
      if c < min
         result ← tmp ∪ {n}
         min ← c
      end if
   end for
   memValue(n) ← result
   return result
```

```
end bestSubSeries


cost(listOfIndices)
    ∑ t(c'_i)|N'_i| for i in listOfIndices
end cost
```

D.3 Application to Timbre Similarity

In this section, we apply the algorithm described above to the practical task of calculating the n nearest neighbor of a song according to a typical timbre similarity measure.

D.3.1 The Precision-Cputime Tradeoff

As seen in Chapter 3.1.3, the prototypical distance compares GMM models using a Monte Carlo approximation of the Kullback-Leibler (KL) distance between each duple of models A and B. The precision of the approximation is clearly dependent on the number of samples drawn from the distributions, which we call Distance Sample Rate (dsr). Figure D.3 (on a semi-logarithmic scale) shows the influence of dsr on the precision of the measure (this result is established in Appendix B.1). We see that the DSR has a positive influence on the precision when it increases from 1 to 2000, and that further increase has little if any influence. Figure D.3 also shows the (rescaled) cpu time profile, which is a linear function of dsr. It appears that first, the algorithm exhibits a precision-cputime tradeoff (using the dsr as tradeoff parameter p), and second that, for small dsr's, the precision of the measure increases faster than its cpu time, which makes it a good candidate for the NN algorithm presented above.

D.3.2 Formulation of the Problem

We apply the algorithm described in Section 3.1 to the task of computing the 100 nearest neighbors of an arbitrary seed song in a database containing 15,554 music files, with respect to the target

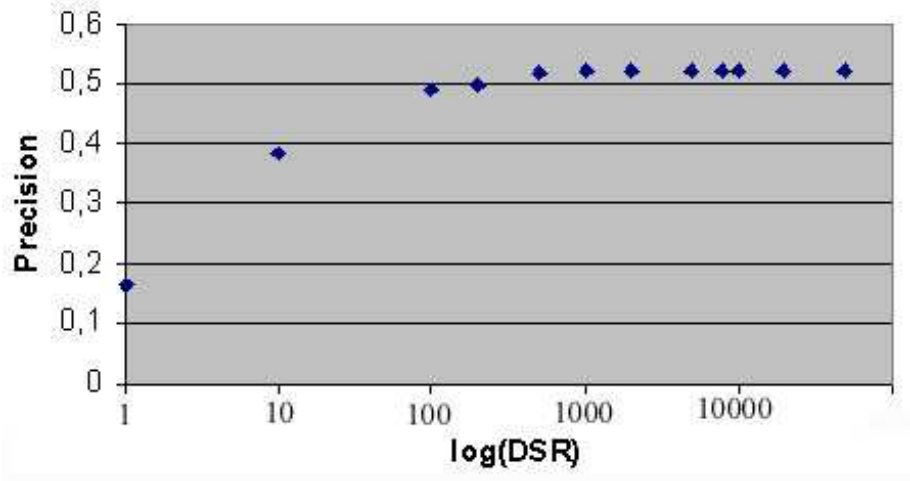

Figure D.3: Influence of the distance sample rate on the precision and cpu time of the timbre similarity algorithm

distance d. In our problem, d is the timbre distance described above using $dsr = 2000$, which is considered to be an ideal setting.

The distance algorithm has a tradeoff parameter $p = dsr$ which takes its integer values in $P = \{1, ..., 2000\}$, and we refer to the instances of the distance which uses p as d_p. Notably, $d = d_{2000}$. The cost of computing d_p is linear in p, and the precision of d_p increases with p.

This problem fits into the scheme presented in Section D.2 if one states it as follows:

- S is the collection of music files
- d_p is the Monte Carlo approximation of the KL distance with p sampling points
- s is an element of S
- $N_p(s)$ is the set of the 100 nearest neighbors of s wrt d_p. In particular, what we want to compute is $N_{2000}(s)$, the set of the 100 nearest neighbors of s wrt $d = d_{2000}$

Given s in S, $\forall i \in \{1, ..., 2000\}$, we define the result sets $N_i \subseteq S$ as follows: $\forall i \in \{1, ..., 2000\}$, N_i is the smallest subset of S such that:

$$\forall x \in N_{2000}, \forall y \in S, d_i(x, s) \geq d_i(y, s) \Rightarrow y \in N_i \tag{D.11}$$

In terms of information retrieval, if we define the set of relevant documents as $N_{2000}(s)$, we can observe that

- $|N_i|$ is the number of documents retrieved by d_i when recall[§] $= 1$, i.e. when we have retrieved all the relevant documents.
- $|N_i|$ is inversely related to the precision[¶] of the measure d_i at recall 1.

$$precision(d_i) = \frac{|N_{2000}(s)|}{|N_i|} = \frac{100}{|N_i|} \tag{D.12}$$

We can now define c_i by:

$$c_i(x) = true \Leftrightarrow x \in N_i(s) \tag{D.13}$$

Let us demonstrate that properties P_1, P_2 and P_3 hold for the c_i thus defined:

- $P1$ is satisfied since the cost of computing d_p is linear in p
- $P2$ and $P3$ are satisfied statistically, since the precision of d_p increases with p and by construction of the N_i result sets.

Therefore, one can apply our approach to the problem of computing N_{2000} for seed song s.

D.3.3 Practical Implementation

In order to find the optimal series of $(c_i)_i$ that minimizes the total cputime of our approach for a given query on $N_{2000}(s)$, we need to estimate the $|N_i|$ for a (large) set of $i \in \{1, ..., 2000\}$. One way to estimate $|N_i|$ is to actually compute the set N_i, i.e.

- apply d_{i-1} on $N_0 = S$ in order to sort the songs in S by distance to s according to d_{i-1}
- find the maximum rank over all songs in N_{2000}. It corresponds to the rank after which all the items of $N_{2000}(s)$ have been retrieved, i.e. $|N_i|$

[§]Recall is the ratio of the number of relevant documents retrieved to the total number of relevant documents in the database.

[¶]The precision is the ratio of the number of relevant documents retrieved to the total number of documents retrieved

However, this direct approach has two major problems.

- The set of $|N_i|$ depends on the seed song, so in theory, we have to apply this procedure for each seed song before being able to find the optimal sequence of steps. This is unpractical, as estimating the $|N_i(s)|$ for a given s is itself longer than the direct calculation of $N_{2000}(s)$ with the standard approach. Moreover, it's a chicken and egg problem, as computing the $|N_i(s)|$ requires to know $N_{2000}(s)$.

- The P distances d_i are stochastic algorithms based on Monte Carlo, which never return the same distance $d_i(s,t)$ between 2 given songs s and t twice (although the variance on the results obviously decreases as dsr increases). Hence, for a given seed song s, the $|N_i(s)|$'s themselves should be averaged over several runs of the above procedure.

To overcome these limitations, we propose to estimate a unique set of $\widetilde{|N_i|}$ for the whole database, by applying the above procedure to a few random songs in the database and averaging the results. This has the drawback that the successive inclusion property (Property P_4) is only statistically verified for the estimated $\widetilde{|N_i|}$, and we have no insurance that, for a given seed song s, at a given step i, the set of items $\widetilde{N_i}$ actually contains all the items in $N_{2000}(s)$. It follows that the final set of items returned by the algorithm after a given series of steps $(c_i)_i$ is only an estimate $\widetilde{N_{2000}(s)}$ of the actual set $N_{2000}(s)$, associated with a precision

$$p((c_i)_i, s) = \frac{|\widetilde{N_{2000}(s)} \cap N_{2000}(s)|}{|N_{2000}(s)|} \tag{D.14}$$

Figure D.4 shows the estimated $\widetilde{|N_i|}$ for $i = 1, ..., 2000$, computed on the test database by averaging the $N_i(s)$ over $n_s = 50$ random songs. The darkest curve corresponds to the average $m = \frac{1}{n_s}\sum_{k=1}^{n_s} |N_i(s_k)|$, the medium curve corresponds to the sum $m + \sigma$ of the average m and the standard deviation σ of the $|N_i(s_k)|$, and the lightest curve to $m + 2\sigma$.

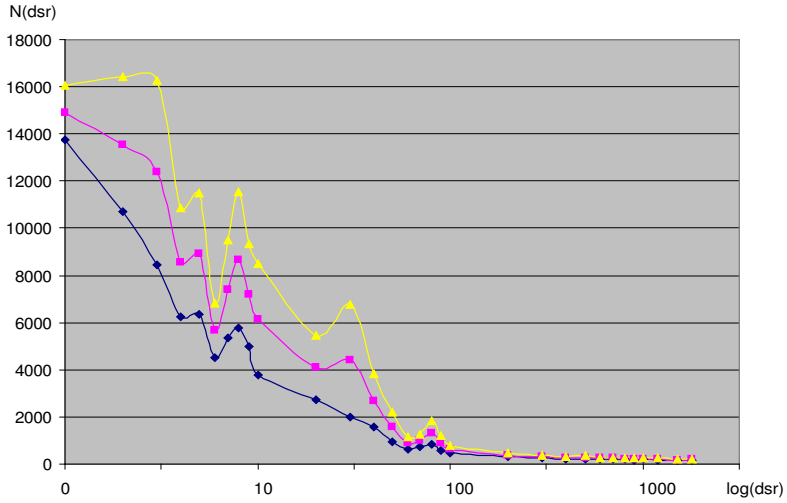

Figure D.4: Convergence profile of the N_i, averaged over 50 NN timbre queries.

Table D.1: Optimal sequences as predicted by dynamic programming

Strategy	**Steps ($\|N_i\|$,i)**	**cost (% standard)**
standard	{15554, 2000}	31,080k (100%)
best (mean)	{15554, 6}, {4501, 20}, {2710, 60}, {652, 200}, {290, 400}, {218, 2000}	1,028k (3.3%)
best (mean - 25%)	{15554, 6}, {3375, 20}, {2032, 60}, {489, 200}, {217, 400}, {163, 2000}	793k (2.6%)
best (mean + σ)	{15554, 6}, {4090, 60}, {894, 200}, {374, 400}, {264, 2000}	1,195k (3.9%)
best (mean + 2σ)	{15554, 6}, {6819, 60}, {1136, 200}, {458, 400}, {310, 2000}	1,532k (4.9%)

D.3.4 Results

We apply our algorithm to the task of calculating the 100 nearest neighbors of a given seed song according to the timbre similarity described above. Table D.1 shows the optimal sequence of steps $(c_i)_i$ obtained with dynamic programming (see Section D.2.3), and the associated cost measured by $\sum_i |N_i| t(c_i)$. We compare the results using the 3 sets of estimated $\widetilde{|N_i|}$ in Figure D.4 and an additional set obtained by downsizing the $|N_i|$ by 25%. For dynamic programming, we make the assumption that the cputime is linear $t(c_i) = \alpha.i + \beta$, with $\alpha = 1$ and $\beta = 0$. It appears that the optimal sequences differ slightly whether we consider the $\widetilde{|N_i|}$ with or without standard deviation. The optimal sequence yields an algorithm which is theoretically more than 30 times faster than the standard approach.

Table D.2 shows the measured performance (cputime and precision) of the actual implementa-

tion of the algorithm for the same sequences of steps. Overall, the cpu performance is very good (we achieve speed improvement factors greater than 30) while still preserving near perfect precision (we retrieve 98% of the 100 true nearest neighbors). We observe that as the $|N_i|$ increase, the precision of the results increases (we are less subjected to accidently pruning relevant nearest neighbors) but also the cpu time. We may observe that the achieved cputime rates are lower than the theoretical predictions (about 1% absolute). This can be explained by the following points :

- The optimal sequence found by dynamic programming and its expected performance were computed using a very simple cpu time model $t(c_i) = i$. This doesn't include e.g. the overhead cost of file I/O (retrieving the GMMs from the database, writing the results, etc.)

- The distance algorithm was not reimplemented to support our recursive approach, i.e. the same executable is run for the successive values of dsr. While this makes the algorithm generic (no need to re-program the distance algorithm it uses), this has an unnecessary cost: each step adds the overhead of its own system call (the executable is called from Java), initialization, file I/O (all the needed GMM files are re-opened at each step, while $|N_{i+1}| - |N_i|$ files are common between each successive call), Gaussian sampling (at each step, dsr_i points are sampled from the Gaussians, while only $dsr_{i+1} - dsr_i$ new points are needed). Most of these overhead costs are not accounted for in the theoretical predictions.

Table D.2: Measured cputime and precision of several sequences of steps $(c_i)_i$

Series	**cpu-time (% stand.)**	**precision**
standard	663.75 (100%)	100%
best (mean)	27.07 (4.0%)	98.2%
best (mean - 25%)	20.98 (3.1%)	94.0%
best (mean + σ)	33.91 (5.1%)	98.9%
best (mean + 2σ)	39.19 (6.0%)	99.0%

Appendix E

Multiscale segmentation

This appendix describes a segmentation algorithm specifically designed for polyphonic music. This algorithm is used in Appendix B.6.2 to synchronize the MFCC extraction frame rate on musical notes, and in Chapter 5 to build a database of polyphonic samples in order to better understand the nature of dynamical modelling.

Typical segmentation algorithms (Tzanetakis and Cook, 1999; Rossignol, 1998) first computes a set of features from the signal cut into frames, and then detect the segment boundaries by looking for abrupt changes in the trajectory of features. Here, we look for the energy variations of the signal. The signal is cut into frames (2048 points at 44100Hz), and for each frame, we compute the short-term spectrum. The spectrum itself is processed by a Mel filterbank of 20 bands. Each band's energy is weighted according to the frequency response of the human ear, as described e.g. in Schroeder et al. (1979). Finally, the energy is summed across all bands. Change detection is done by smoothing the energy profile by a zero-phase filtering by a Hanning window of size S_w, and looking for all the local maxima of the smooth version. The segment boundaries are the deepest valleys in the raw energy profile between 2 adjacent peaks in the smooth profile.

While this scheme is effective for simple, percussive music, we observe that for non percussive, richer polyphonic music, the quality of the segmentation depends on the choice of S_w. In large events such as a sung note lasting for several seconds (e.g. the final "-day" in "Yesterday"), there

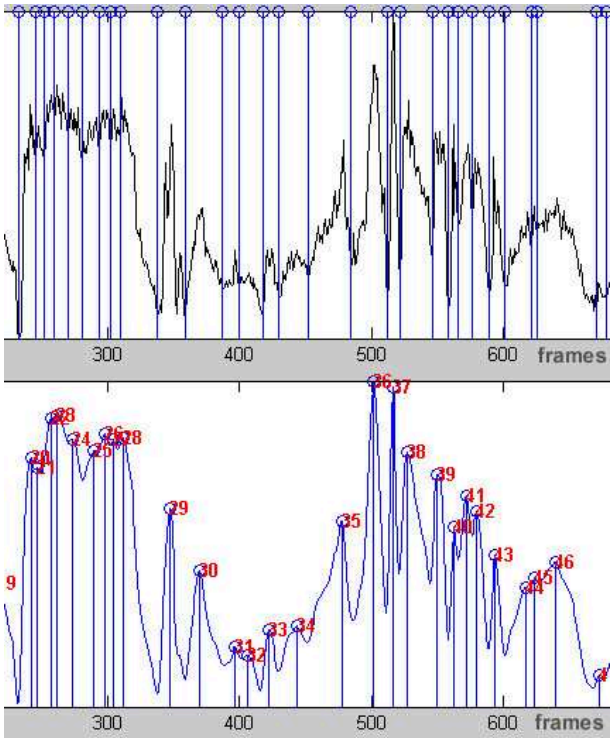

Figure E.1: Segmentation of an extract of *The Beatles - Yesterday*. (Top) Segmented energy profile using a small S_w (150ms) : short events (right) get properly detected, while larger events (left) get oversegmented. (Bottom) Corresponding smoothed energy profile, used for peak detection.

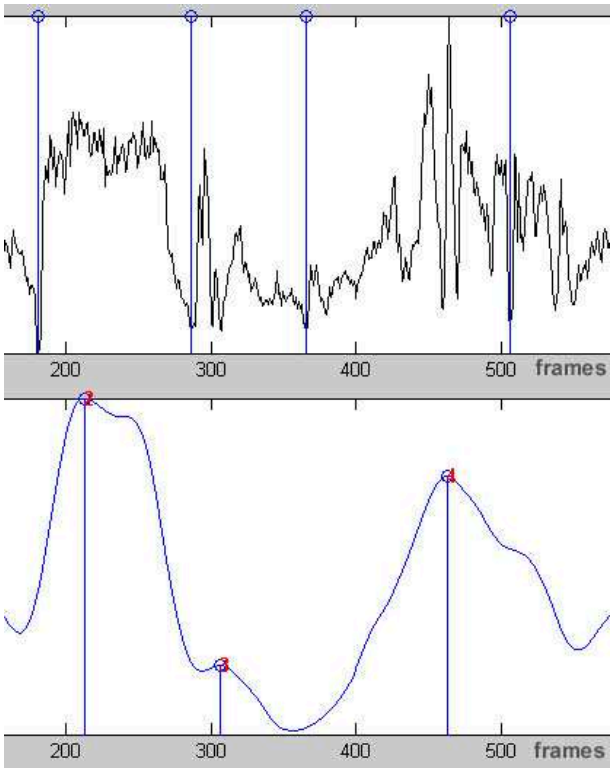

Figure E.2: Segmentation of an extract of *The Beatles - Yesterday*. (Top) Segmented energy profile using a large S_w (1s) : large events (left) are appropriately recognized, however smaller events (right) are missed out. (Bottom) Corresponding smoothed energy profile

may be several small peaks of energy corresponding to the other instruments playing in the background (e.g. a succession of chords played on the guitar). With a small S_w, all these peaks would be segmented, and the most meaningful atomic event would be cut into several short identical notes (see Figure E.1). With a large S_w on the other hand, short meaningful events like isolated guitar chords get missed out (Figure E.2).

Therefore we propose a multiscale segmentation algorithm, which adapts the size of the convolution window to the local shape of the energy profile. More precisely, we compute the STFT of the energy profile on a running 2-second window (with 90% overlap). As the energy profile is sampled using 50 overlapping, 2048 point frames (i.e. 43Hz), the FFT describes the frequency content between 0 and 20Hz, with a frequency resolution finer than 1Hz. We select the predominant local periodicity of the profile as the barycentre point (spectral centroid) of the spectral distribution

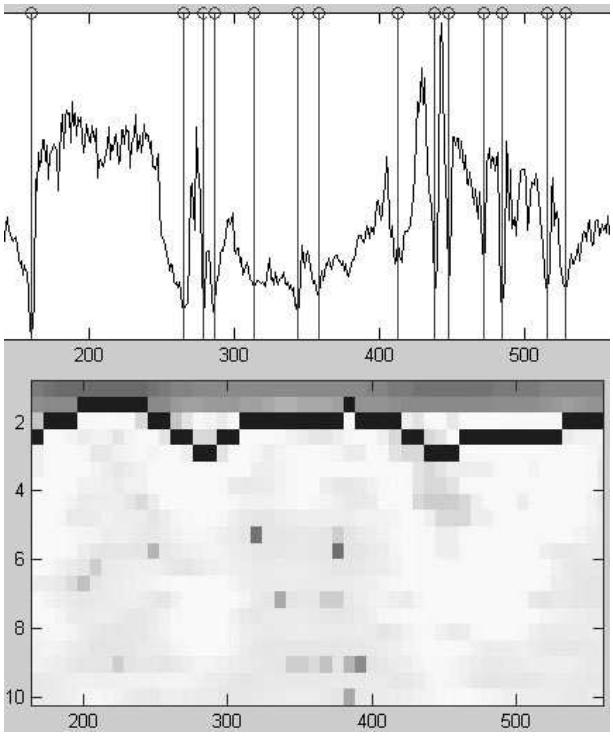

Figure E.3: (Top) Multiscale segmentation of the same extract, using an adaptive convolution window size: large windows on the left, and smaller windows on the right. (Bottom) Corresponding spectrogram of the energy profile, super-imposed (in black) with the spectral centroid of each frame, used to determine the windows size

within each frame :

$$sc = \frac{\sum_k kS(k)}{\sum_k S(k)} \tag{E.1}$$

where S is the magnitude spectrum of a frame. We then smooth the energy profile using a Hanning window size S_w equal to the inverse of the centroid of the corresponding FFT frame (to ensure continuity, Hanning window coefficients are normalized so they sum to one regardless of their length).

Figure E.3-Bottom shows the SFFT of the energy profile used in Figure1. Large events correspond to low frequencies in the energy profile, i.e. small centroid frequencies in the spectrogram (order of 1Hz). Consequently, these zones get smoothed with large Hanning windows (order of 1 sec.). On the other hand, short events in the energy profile correspond to higher frequency content, higher centroids, and smaller windows size (order of 200ms). Figure E.3-Top illustrates the

corresponding multiscale segmentation, which preserves large, noisy events as well as short, high amplitude ones.

Appendix F

Measures of Hubs

Several measures can be used to identify and quantify the "hubness" of a given song. In Chapter 6, two of such measures are used: number of occurrences and mean neighbor angle. We give here complete details on these measures, as well as a number of alternatives, and compare them to one another. Notably, we show that hub songs are not typically correlated with violations of the triangular inequality.

F.1 Rank-based metrics

A natural measure of the hubness of a given song is the number of times the song occurs in the first n nearest neighbors of all the other songs in the database. It is easily computed on a sparse version of a similarity matrix (see Chapter 4) by counting the number of duples containing the song stored in the matrix. Table 6.1 in Chapter 6 shows a few songs in the test database along with the number of times they occur in the first 10 nearest neighbors over all queries (N_{10}). This illustrates the predominance of a few songs that occur very frequently. For instance, the first song, `MITCHELL, Joni - Don Juan's Reckless Daughter` is very close to 1 song out of 6 in the database (57 out of 350).

As was already summarized in Chapter 6, the total number of occurrence of a song has a number

of properties:

- Independent of distance: Being based on rank, the number of occurrences of a song is independent of the range of the values produced by a given distance measure. Therefore, it can be used to compare hubs appearing with different algorithms, which we will do e.g. in Section 6.6.

- Dependant on database: The total number of occurrence of a song is composed both of true and false positives. As explained earlier, only the latter are characteristic of a hub. This metric therefore is conservative in the sense that if a high number of occurrence is observed for a given song in an arbitrary database, it is difficult to conclude whether it is indeed a hub (i.e. that most of these occurrences correspond to false positives) or a perceptual center-of-mass (i.e. most of the occurrences are true positives).

  A way to compensate this limitation is to use a ground truth when available. Table 6.1 compares the N_{10} to the size of the cluster of each song ($card(C_S)$). One can see e.g. that `Don Juan...` occurs more than 6 times more than it should (e.g. is close to 6 times more songs than the number of relevant neighbors identified by the ground truth). As our test database has been designed with songs of a large variety of genres and periods, it is reasonable to assume that many of the occurrences of `Don Juan...` are likely to be false positives. However, such a ground truth normalization is limited, since perceptually relevant matches could occur across different clusters. For instance, "rock" songs from cluster "The Clash" are likely to occur in the close matches of more songs that the mere Clash songs, since songs of other clusters have related timbres, such as "Brian Adams" or "Gary Moore". Such inter-cluster true positives cannot be identified using our ground truth, and cannot be counted to compensate the raw number of occurrences.

- Constant-sum: An important property of the number of n-occurrences N_n of a song is that the sum of the values for all songs is constant given a database. Each query only gives the opportunity for n occurrences to the set of all the other songs, such that the total number of

n-occurrences in a given $\mathcal{N}$-size database is $n * \mathcal{N}$. Therefore, the mean n-occurrence of a song is equal to n, independently of the database and the distance measure. Alternatively, if we assume that the distance engenders a uniform, random distribution of the song, a given song has the probability $p = \frac{n}{\mathcal{N}}$ to occur in the n-nearest neighbor of another song, which indeed gives an expected number of occurrences $E(N_n) = \mathcal{N} * p = n$. Table F.1 illustrates the experimental verification of this property (constant mean) for several distance algorithms.

Table F.1: Comparison of mean number of occurrences and mean neighbor angle for songs in the test database, for several distance algorithms

Measure	GMM	HMM	Delta	Acceleration	Histogram
N_{100}	100.2	98.7	99.4	99.4	99.6
Neighbor Angle (degrees)	58.8	55.6	58.3	57.9	59.9

- Descriptive statistics: This has the notable consequence that the mean value of N_n is useless to measure the influence of a given algorithm on the global hubness of a database. One has to look for other descriptive statistics, such as the variance of the distribution of occurrences, or the number of songs with more than a given number of occurrences.

F.2 Distance-based metrics

An operational definition of a hub is that it is a song H which is found to be "close" (though not perceptually) to duplets of songs A and B which themselves are (perceptually) distant from one another. Note that songs close to many songs which are themselves close to one another would indicate an acceptable "center-of-mass" situation. Therefore, the hubness of song H can be estimated by comparing its distances to its neighbors $d(H, A)$ and $d(H, B)$ on the one hand, and the distance between the neighbors $d(A, B)$ on the other hand. We propose 3 metrics using this idea:

- Neighbor difference: This measures the difference between the neighbor distance and the mean distance to the neighbors, using

$$h_1(H, A, B) = d(A, B) - \frac{d(H, A) + d(H, B)}{2} \tag{F.1}$$

This is computed for a given song H by drawing a large number of successive duplets (A, B) (such that $A \neq B \neq H$) where A and B are close neighbors of H (typically in the first n nearest neighbors), and computing the mean value of $h_1(H, A, B)$. We use 1000 successive random draws.

- Neighbor angle: This measures the angle θ formed by the segments $[H, A]$ and $[H, B]$. This attempts to normalize the previous measure by the actual amplitude of the distances. As seen in Figure F.1, the angle θ can be expressed in terms of $d(H, A)$, $d(H, B)$ and $d(A, B)$.

$$d = d(H, B)\sin\theta = d(A, B)\sin\alpha \tag{F.2}$$

$$d' = d(H, A) - d(H, B)\cos\theta = d(A, B)\cos\alpha \tag{F.3}$$

$$\Rightarrow d^2 + d'^2 = d(H, A)^2 + d(H, B)^2 - 2d(H, A)d(H, B)\cos\theta = d(A, B)^2 \tag{F.4}$$

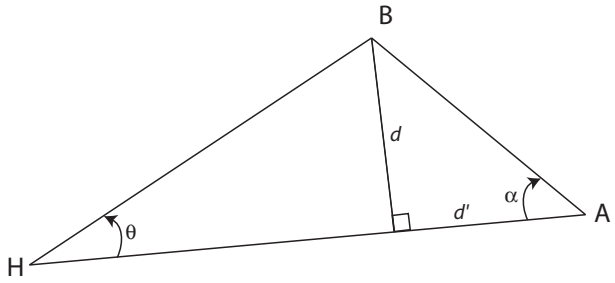

Figure F.1: The neighbor angle θ can be expressed in terms of $d(H, A)$, $d(H, B)$ and $d(A, B)$.

and therefore

$$h_2(H, A, B) = \cos\theta = \frac{d(A, B)^2 - d(H, A)^2 - d(H, B)^2}{2d(H, A)d(H, B)} \tag{F.5}$$

As before, this is computed for a given song H by drawing a large number of successive duplets of neighbors (A, B) (such that $A \neq B \neq H$), and computing the mean value of $h_2(H, A, B)$. We use 1000 successive random draws.

- Triangular Inequality Violation: A final measure based on the ratio of distances to neighbors

examines whether the distances to a given point violate the triangular inequality (TI), i.e.

$$d(H, A) + d(H, B) < d(A, B) \tag{F.6}$$

This is of course impossible for proper mathematical distances, such as e.g. euclidean distance or Kullback-Leibler divergence. However, many of the metrics considered in Chapter 5, and notably the best-performing one, are not mathematical distances and have no guarantee to respect TI. In the next section, we will examine whether potential violations of TI indeed correspond to hub songs. As before, we draw a large number of successive duplets of neighbors (A, B) (such that $A \neq B \neq H$), and estimate the probability of violating the TI. We use 1000 successive random draws.

Note that for all three measures, we notably assume that the distance between neighbors $d(A, B)$ is always perceptually relevant, i.e. that for instance, A and B are not hubs themselves. This only holds statistically, to the amount of precision of the examined distance measure (e.g. around 70% for the best algorithms).

Distance-based metrics have the following properties:

- Independent of database: Unlike measures of the number of occurrence of a song, distance-based metrics are independent of the possible perceptual clusters of a given database. Thus they can be used to compare algorithms on different databases.

- Dependant on algorithm: However, distance-based metrics are typically dependent on the distance algorithm. Neighbor difference is trivially dependent of the range of the values. Neighbor angle, although independent from the actual amplitude of the distance values (by normalization), is still dependent on the discrimination capacity of the distance, i.e. the typical distance ratio between what can be considered a close distance, and what can be considered a large distance. This will be detailed in Section F.3.3, where we will see that this also influences the measure of TI-violation.

- Constant-sum: An important property of the neighbor-angle value is that, like the number of n-occurrences N_n of a song, the sum of the values for all songs is constant given a database size. This directly derives from the fact that the angles of a triangle sum to π radians (in a euclidean geometry - which is only approximated here in the general case). Given a set of $\mathcal{N}$ points, the number of angles whose vertex is a given point X, and are formed by the lines from X to the $\mathcal{N}-1$ other points, is equal to the number of combinations of 2 points within $\mathcal{N}-1$, i.e. $C^2_{\mathcal{N}-1}$. There are $\mathcal{N}$ possible vertex X for such angles, thus there are a total of $\mathcal{N}C^2_{\mathcal{N}-1} = \frac{n(n-1)(n-2)}{2}$ angles formed between the $\mathcal{N}$ points. It is easy to see that $n(n-1)(n-2)$ is divisible by $3 \forall n$. Hence, these angles can be clustered by triplets, so that their supporting lines form a triangle, and thus sum to π. Therefore, the sum of all angles formed between $\mathcal{N}$ points equals $\frac{\mathcal{N}C^2_{\mathcal{N}-1}}{3}\pi$. Table F.1 illustrates the experimental verification of this property (constant mean) for several distance algorithms. The deviation of the mean angle from the theoretical value $60°$ is both explained by the statistical approximation of the computation of the angles and by the possible non-euclideanity of the underlying geometry.

- Descriptive statistics: This has the notable consequence that the mean value of the Neighbor Angle is useless to measure the influence of a given algorithm on the global hubness of a database. Like for occurrence values, one has to look for other descriptive statistics, such as the number of songs with a mean angle greater than a given limit.

F.3 Correlation between measures

We examine in this section the correlations between the various hubness measures proposed above. We report the measures on the same set of algorithms that was used in Chapter 6.6.

F.3.1 Number of N-occurrences

Table F.2 show poor linear correlation between number of occurrences for varying values of N. However, Figure F.2 is typical of the kind of scatter plot obtained when one component is part of

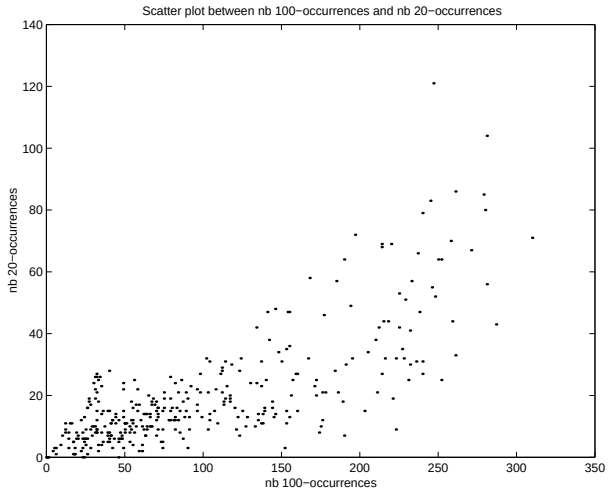

Figure F.2: Scatter plot between number of 100-occurrences N_{100} and number of 20-occurrence N_{20} for a distance based on HMM.

the other: if one increases the sum, then the parts also increases. The precise relation between N_{100} and N_{20} depends on the composition of the database, and notably the size of its clusters. If the database tends to be very densely clustered (e.g. 20 heavy metal songs and 20 symphonic pieces), then the correlation between values of N larger than the mean cluster size (say 20) will be high.

Table F.2: Correlation between number of 100-occurrences N_{100} and number of 20-occurrence N_{20} for various models

GMM	HMM	Delta	Acceleration	Histogram
0.66	0.76	0.75	0.77	0.77

F.3.2 Number of N-occurrences and Neighbor difference and angle

As can be seen in Figures F.3 and F.4, there is a nearly logarithmic dependency between the number of occurrence of a given song and both its mean Neighbor difference and angle. Table F.3 shows the linear correlation scores between the logarithm of N_{100} and both measures. The best fits are achieved for the static models, both parametric and non-parametric. Dynamic models tend to create more outlier points in the scatter plots, which reduce the correlation scores. It appears nevertheless that hub songs tend to be associated to higher values of Neighbor difference and angle.

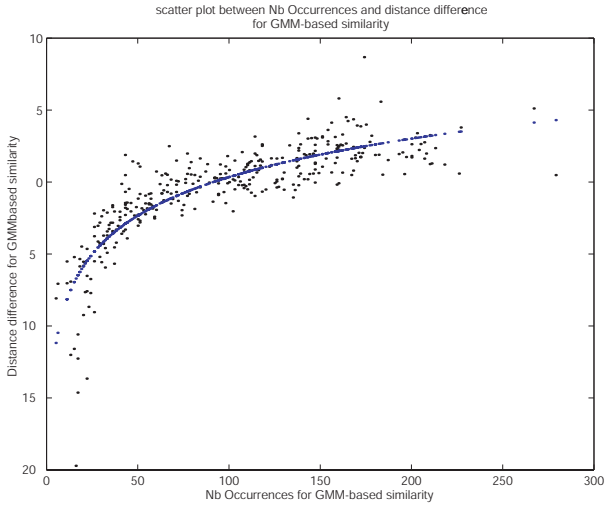

Figure F.3: Scatter plot between number of 100-occurrences N_{100} and Mean Neighbor Difference scores for a distance based on GMM.

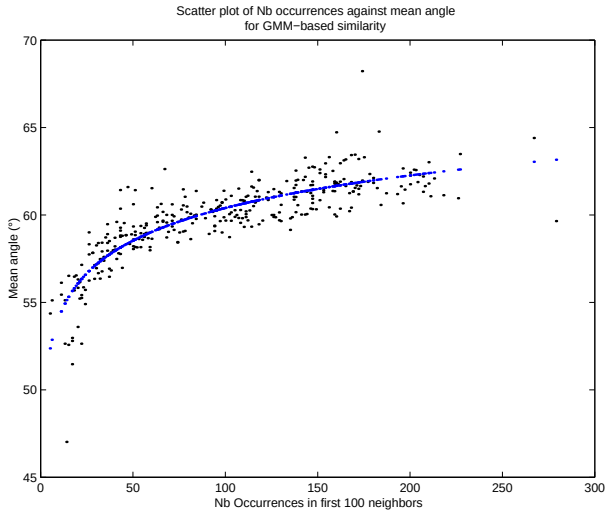

Figure F.4: Scatter plot between number of 100-occurrences N_{100} and Mean Neighbor Angle for a distance based on GMM.

Table F.3: Correlation between the logarithm of the number of 100-occurrences N_{100} on the one hand, and neighbor difference and mean angle on the other hand, for various models

Measure	GMM	HMM	Delta	Acceleration	Histogram
Neighbor difference	0.85	0.78	0.66	0.68	0.87
Neighbor angle	0.85	0.76	0.73	0.74	0.93

F.3.3 Number of N-occurrences and TI violations

Table F.4 shows the linear correlation scores between N_{100} and the probability of violation of triangular inequality for all 5 algorithms. Little correlation (if any) can be found in most cases, with the exception of a limited linear fit for HMM-based measures. The GMM-based distance creates very few TI violations, but however still exhibit some severe hubs, which creates null correlation. The same is true obviously for Histogram-based distance, which use euclidean distance, and thus do not violate TI by construction. However, HMM-based measures seem to create quite a few TI violations. These necessarily correspond to high values of neighbor difference and angle, and thus are associated to high values of nb occurrences (since there is correlation in between the latter). Delta and Acceleration-based metrics create a few TI violations, however not sufficiently so maybe that a clear correlation tendency can be observed with N_{100}.

Table F.4: Correlation between the number of 100-occurrences N_{100}, and TI violation probability on the other hand, for various models

GMM	HMM	Delta	Acceleration	Histogram
0.0	0.74	0.35	0.17	0.0

In any case, TI violation probability appears to be a poor metric for hub quantification: hubs can appear without TI violation. This can be explained by the fact that the timbre distances typically have poor discrimination power. Figure F.5 shows the distributions of the distances of all songs to songs in the clusters “The Clash” and “Accordion Musette”. It appears that distances values have a relatively small range ($[75, 95]$), with a majority of values in the range $[80, 85]$. Therefore, even if 2 neighbors A and B are “optimally” close to a given song H (with e.g. $d(A, H) = d(B, H) = 75$), and that these neighbors are “optimally” distant (with e.g. $d(A, B) = 95$), TI can be preserved while

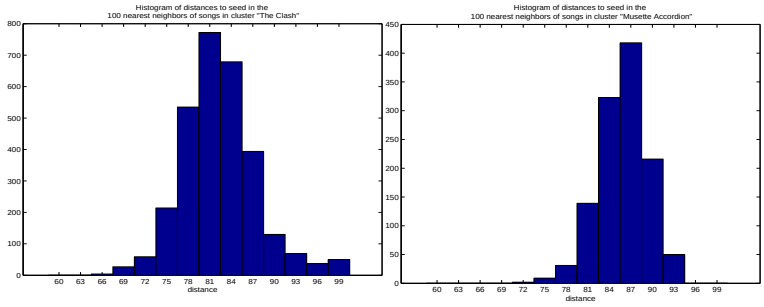

Figure F.5: Distribution of the distances to songs of the cluster “The Clash” (left) and “Musette Accordion” (right), with a GMM-based distance.

H clearly behaves like a hub. The different shapes observed for the clusters in Figure F.5 reveal the composition of the test database. As already noted, a few clusters like “Gary Moore”, “Brian Adams”, “Jimi Hendrix” contain songs that are timbrally related (and could be gathered under the umbrella term “rock”). Therefore, the songs in cluster “The Clash” exhibit small distances not only to songs of the same cluster, but also to a good number of songs from the rock clusters. This explains that the “Clash” histogram contains a large proportion of “small” distances in the range $[75, 85]$. On the other hand, cluster “Musette Accordion” is a pretty isolated group in the test database, and therefore songs in this cluster are typically quite distant to songs from other clusters. This explains that the “Musette” histogram contains a minority of “small’ distances, and a majority of larger distances in the range $[85, 95]$. For comparison purpose, Figure F.6 shows the distribution of the distances to songs that exhibit a number of occurrences larger than 180 (out of a maximum 360). One can see that such hub songs have a majority of small distances to other songs in the database, in the range $[75, 85]$.

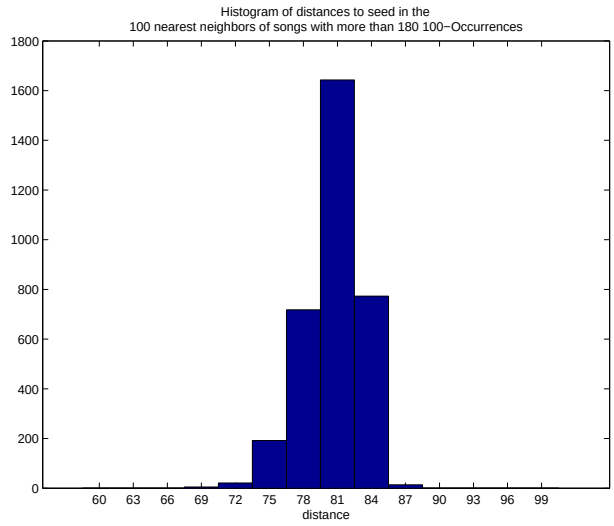

Figure F.6: Distribution of the distances to songs that exhibit a number of 100-occurrences larger than 180 (out of 360 in the test database), with a GMM-based distance.

Appendix G

Pearson's χ^2-test of independence

We describe here Pearson's χ^2-test used to assess the statistical independence of pairs of attributes in Chapter 7. χ^2 tests the hypothesis (called the *null hypothesis*) that the relative frequencies of occurrence of observed events follow a flat random distribution (e.g. that hard rock songs are not significantly more likely to talk about violence than non hard-rock songs). χ^2 is calculated by finding the difference between each observed and theoretical frequency, squaring them, dividing each by the theoretical frequency, and taking the sum of the results:

$$\chi^2 = \sum \frac{(O - E)^2}{E} \tag{G.1}$$

where O is an observed frequency and E an expected (theoretical) frequency, asserted by the null hypothesis.

For example, to test the hypothesis that the two attributes "TextCategory Violence" and "Style Metal" are independent variables, we use the contingency table shown in Table G.1. The figures in the right-hand column and the bottom row are called marginal totals and the figure in the bottom right-hand corner is the grand total.

If the 2 attributes were independent, the theoretical count of, say, violent and metal songs would

Table G.1: Contingency Table for "TextCategory Violence" and "Style Metal". Values in parenthesis are the theoretical counts under the independence hypothesis.

	Violent	Non violent	Total
Metal	36 (7)	122 (152)	158
Non metal	180 (209)	4598 (4569)	4778
Total	216	4720	4936

be

$$N_{total}p(metal\&violent) = N_{total}p(metal)p(violent) = 4936\frac{158}{4936}\frac{216}{4936} = 6.91 \tag{G.2}$$

Table G.1 shows such expected counts in parenthesis. The χ^2 value for Table G.1 is therefore computed as:

$$\chi^2 = \frac{(36-7)^2}{7} + \frac{(122-152)^2}{122} + \frac{(180-209)^2}{180} + \frac{(4598-4569)^2}{4598} = 132.36 \tag{G.3}$$

If the null hypothesis is true, the χ^2 test follows the probability distribution (called the χ^2 distribution) of the random variable $X = Z_1^2 + Z_2^2$ where the Z_i are independent standard normal variables (zero expected value and unit variance). This distribution can be used to compute the probability of observing the counts in Table G.1 if metal and violent contents were independent, which in our case is less than 0.1%. This probability is lower than conventional criteria for statistical significance, so normally we would reject the null hypothesis that the 2 attributes are independant. The degrees of association between duplets of variables are usually assessed and compared by a number of coefficients which are independant from the size of the population and the number of attributes being compared. The simplest is the Φ coefficient defined by

$$\Phi = \sqrt{\frac{\chi^2}{N}} \tag{G.4}$$

where χ^2 is derived from the Pearson test, and N is the grand total number of observations. Φ varies from 0 (corresponding to no association between the variables) to 1 (complete association).

Bibliography

Bibliography

Albert, R., Jeoung, H., and Barabasi, A.-L. (1999). The diameter of the world wide web. *Nature*, 401:130.

ASA (1960). Acoustical terminology, s.1.1-1960. american standards association.

Aucouturier, J.-J., Aurnhammer, M., and Pachet, F. (2006a). Sound textures: An investigation of image texture analysis for audio timbre similarity. *IEEE Transactions on Speech and Audio Processing (submitted)*.

Aucouturier, J.-J., Defreville, B., and Pachet, F. (2006b). The bag-of-frame approach to audio pattern recognition: Why this works for urban soundscapes and not for polyphonic music. *Journal of the Acoustical Society of America (submitted)*.

Aucouturier, J.-J. and Pachet, F. (2002a). Finding songs that sound the same. In *Proceedings of the 1st IEEE Benelux Workshop on Model Based Processing and Coding of Audio (MPCA), Leuven (Belgium)*. Invited Talk.

Aucouturier, J.-J. and Pachet, F. (2002b). Music similarity measures: What's the use ? In *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR), Paris (France)*.

Aucouturier, J.-J. and Pachet, F. (2002c). Scaling up music playlist generation systems. In *Proceedings of The IEEE International Conference on Multimedia and Expo, Lausanne (Switzerland)*.

Aucouturier, J.-J. and Pachet, F. (2003). Representing musical genre: A state of art. *Journal of New Music Research (JNMR)*, 32(1).

Aucouturier, J.-J. and Pachet, F. (2004a). Improving timbre similarity: How high's the sky ? *Journal of Negative Results in Speech and Audio Sciences*, 1(1).

Aucouturier, J.-J. and Pachet, F. (2004b). Tools and architecture for the evaluation of similarity measures: Case study of timbre similarity. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, Barcelona, Spain.

Aucouturier, J.-J. and Pachet, F. (2005). Ringomatic: A real-time interactive drummer using constraint-satisfaction and drum sound descriptors. In *Proceedings of the International Conference on Music Information Retrieval*. London.

Aucouturier, J.-J. and Pachet, F. (2006a). The influence of polyphony on the dynamical modelling of musical timbre. *Elsevier Pattern Recognition Letters (submitted)*.

Aucouturier, J.-J. and Pachet, F. (2006b). Jamming with plunderphonics: Interactive contatenative synthesis of music. *Journal of New Music Research*.

Aucouturier, J.-J. and Pachet, F. (2006c). A scale-free distribution of false positives for a large class of audio similarity measures. *IEEE Transactions on Pattern Analysis and Machine Intelligence (submitted)*.

Aucouturier, J.-J., Pachet, F., and Hanappe, P. (2004). From sound sampling to song sampling. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, Barcelona, Spain.

Aucouturier, J.-J., Pachet, F., and Sandler, M. (2005). The way it sounds: Timbre models for analysis and retrieval of polyphonic music signals. *IEEE Transactions on Multimedia*, 7(6):1028–1035.

Aucouturier, J.-J. and Sandler, M. (2001a). Segmentation of musical signals using hidden markov models. In *Proceedings of the 110th Convention of the Audio Engineering Society, Amsterdam (The Netherlands)*.

Aucouturier, J.-J. and Sandler, M. (2001b). Using long-term structure to retrieve music: Representation and matching. In *Proceedings of the 2nd International Symposium on Music Information Retrieval (ISMIR), Bloomington, Indiana (USA)*.

Aucouturier, J.-J. and Sandler, M. (2002). Finding repeating patterns in acoustical musical signals. In *Proceedings of the AES 22th International Conference on Virtual, Synthetic and Entertainment Audio, Espoo, Finland*.

Bak, P. (1996). *How Nature Works: The science of self-organized criticality*. Springer-Verlag, New York.

Baumann, S. (2003). Music similarity analysis in a p2p environment. In *Proceedings of the 4th European Workshop on Image Analysis for Multimedia Interactive Services, London (UK)*.

Baumann, S. and Pohle, T. (2003). A comparison of music similarity measures for a p2p application. In *Proceedings of the Sixth International Conference on Digital Audio Effects DAFX, London (UK)*.

Baumann, S., Pohle, T., and Shankar, V. (2004). Towards a socio-cultural compatibility of mir systems. In *Proceedings of the Fifth International Conference on Music Information Retrieval, Barcelona, Spain*.

Berenzweig, A., Logan, B., Ellis, D. P. W., and Whitman, B. (2003). A large-scale evaluation of acoustic and subjective music similarity measures. In *Proceedings of the Fourth International Conference on Music Information Retrieval (ISMIR 2003), Baltimore, Maryland (USA)*.

Berger, H. and Fales, C. (2003). The match of perceptual and acoustic features of timbre over time: "heaviness" in the perception of heavy metal guitar textures. Retrieved 10/01/2006 from `www.indiana.edu/~savail/workingpapers/heavy.html`).

Bimbot and Chollet (1997). *Assessment of speaker verification systems.*, chapter 11. Mouton de Gruyter, Berlin.

Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford Press.

Blumer, A., Ehrenfeucht, A., Haussler, D., and Warmuth, M. K. (1989). Learnability and the vapnik-chervonenkis dimension. *Journal of the ACM*, 36(4):865–929.

Borg, I. and Groenen, P. (1997). *Modern Multidimensional Scaling.* Springer-Verlag, New York.

Bozkaya, T. and Ozsoyoglu, M. (1999). Indexing large metric spaces for similarity search queries. *ACM Transactions on Database Systems*, pages 1–34.

Brown, J. c. (1997). Computer identification of musical instruments using pattern recognition with cepstral coefficients as features. *Journal of Acoustical Society of America*, 105(3):1933–1941.

Bruno, N., Chaudhuri, S., and Gravano, L. (2001). Stholes: a multidimensional workload-aware histogram. In *Proceedings of the 2001 ACM SIGMOD international conference on Management of data, Santa Barbara, California, United States*.

Cano, P. and Koppenberger, M. (2004). The emergence of complex network patterns in music artist networks. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, Barcelona, Spain.

Casagrande, N., Eck, D., and Kégl, B. a. (2005). Frame-level speech/ music discrimination using adaboost. In *Proceedings of International Symposium of Music Information Retrieval, London (UK)*.

Casey, M. and Crawford, T. (2004). Automatic location and measurements of ornaments in audio recordings. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR 2004)*.

Ciacca, P., Patella, M., and Zezula, P. (1997). M-tree: an efficient access method for similarity search in metrix spaces. In *Proceedings of the 23rd International Conference on Very Large Databases*.

Clarkson, B., Pentland, A., and Mase, K. (2000). Recognizing user context via wearable sensors. In *Proceedings of the 4th IEEE International Symposium on Wearable Computers*, page 69, Washington, DC, USA.

Cole, R. A. E., Mariani, J., Uszkoreit, H., Zaenen, A., and Zue, V. (1996). *Survey of the State of the Art in Human Language Technology*. Cambridge University Press.

Collobert, R., Bengio, S., and Mariéthoz, J. (2002). Torch: a modular machine learning software library. Technical Report IDIAP-RR 02-46, IDIAP.

Cramér, H. (1957). *Mathematical Methods of Statistics*. Princeton University Press, Princeton.

Crochemore, M. and Rytter, W. (1994). *Text Algorithms*. Oxford University Press.

Daudet, L. (2006). Sparse and structured decompositions of signals with the molecular matching pursuit. *IEEE Transactions on Speech and Audio Processing*. (in press).

Delalande, F. (2001). *Le Son des Musiques, entre Technologie et Esthtique*. INA - Buchet/Chastel, Paris.

DePoli, G. and Prandoni, P. (1997). Sonological models for timbre characterization. *Journal of New Music Research*, 26(2).

Deshpande, H., Nam, U., and Singh, R. (2001). Classification of music signals in the visual domain. Proceedings of DAFX01.

Dhanaraj, R. and Logan, B. (2005). Automatic prediction of hit songs. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, London, UK.

Doddington, G., Liggett, W., Martin, A., Przybocki, M., and Reynolds, D. (1998). Sheep, goats, lambs and wolves, a statistical analysis of speaker performance. In *Proceedings of the 5th International Conference on Spoken Language Processing (ICSLP), Sydney (Australia)*.

Dubnov, S. and Fine, S. (1999). Stochastic modeling and recognition of solo instruments using harmonic and multi band noise features. Technical report, HUJI Israel. available from www.cs.huji.ac.il/labs/learning/Papers/ icmc99.txt.

Ellis, D. (2005). Re: [music-ir] human and automatic genre classification. Email answer to Arthur Flexer, posted Mon, 26 Sep 2005 11:11:51 -04 on the music-ir mailing list.

Ermolinskiy, A., Cook, P., and Tzanetakis, G. (2001). Musical genre classification based on the analysis of harmonic content features in audio and midi. Princeton University Work Report.

Eronen, A. (2001). Comparison of features for musical instrument recognition. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA), Mohonk, New-Paltz, NY.*

Eronen, A. (2003). Musical instrument recognition using ica-based transform of features and discriminatively trained hmms. In *Proceedings of the Seventh International Symposium on Signal Processing and its Applications (ISSPA), Paris, France*, pages 133–136.

Eronen, A. and Klapuri, A. (2000). Musical instrument recognition using cepstral coefficients and temporal features. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Istanbul (Turkey)*.

Essid, S., Richard, G., and David, B. (2004). Efficient musical instrument recognition on solo performance music using basic features. In *Proc. of the AES 25th International Conference, London, UK*.

Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database*. MIT PRess.

Fishman, G. S. (1996). *Monte Carlo: Concepts, Algorithms and Applications*. Springer Series in Operations Research.

Flexer, A., Pampalk, E., and Widmer, G. (2005). Novelty detection based on spectral similarity of songs. In *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR), London, UK*.

Foote, J. (2000). Automatic audio segmentation using a measure of audio novelty. In *Proceedings of IEEE International Conference on Multimedia and Expo (ICME), New York City, NY (USA)*.

Foote, J. T. (1997). Content-based retrieval of music and audio. In *Multimedia Storage and Archiving Systems II, Proc. of SPIE*. pages 138-147.

Foote, J. T. and Silverman, H. F. (1994). A model distance measure for talker clustering and identification. In *Proceedings ICASSP '94*, pages 317–320.

Francès, R. (1988). *The Perception of Music (translated by W. Jay Downling)*. Lawrence Erlbaum Associates, Hilsdale, New Jersey.

Freedman, D., Pisani, R., and Purves, R. (1997). *Statistics, 3rd edition*. W.W. Norton and Co., New York.

French, J. C. and Hauver, D. B. (2001). Flycasting: On the fly broadcasting. In *Proc. WedelMusic Conference, Firenze, Italy*.

Fujinaga, I. (1998). Machine recognition of timbre using steady-state tone of acoustical musical instruments. In *Proceedings of the International Computer Music Conference (ICMC)*, pages 207–210.

Furui, S. (1986). Speaker-independent isolated word recognition using dynamic features of speech spectrum. *IEEE Transactions on Speech and Audio Processing*, 34(1):52–59.

Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1995). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional Computing Series.

Georgiev, P. G., Theis, F., and Cichocki, A. (2005). Sparse component analysis and blind source separation of underdetermined mixtures. *IEEE Transactions on Neural Networks*, 16(4):992–996.

Ghias, A., Logan, J., Chamberlin, D., and Smith, B. C. (1995). Query by humming – musical information retrieval in an audio database. In *Proceedings of ACM Multimedia, San Francisco, California*.

Gómez, E., Klapuri, A., and Meudic, B. (2003). Melody description and extraction in the context of music content processing. *Journal of New Music Research*, 32(1).

Gouyon, F. and Dixon, S. (2004). Dance music classification: a tempo-based approach. In *Proceedings of the 5th International Conference on Music Information Retrieval*, Barcelona, Spain.

Grey, J. M. (1977). Multidimensional perceptual scaling of musical timbres. *Journal of the Acoustical Society of America*, 61:1270–1277.

Hanna, P. and Desainte-Catherine, M. (2003). Analysis method to approximate the spectral density of noises. In *Proceedings of the 2005 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA), Mohonk, New-Paltz, NY.*

Haralick, R. M., Shanmugam, K., , and Dinstein, I. (1973). Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics*, pages 610–621.

Heittola, T. and Klapuri, A. (2002). Locating segments with drums in music signals. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*.

Herre, J., Allamanche, E., and Ertel, C. (October 2003). How similar do songs sound? towards modeling human perception of musical similarity. In *Proceedings of IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA), Mohonk, NY (USA)*.

Herrera-Boyer, P., Peeters, G., and Dubnov, S. (2003). Automatic classification of musical instrument sounds. *Journal of New Music Research*, 32(2):3–21.

Hicklin, A., Watson, C., and Ulery, B. (2005). The myth of goats: How many people have fingerprints that are hard to match? (patriot act report 7271), National Institute of Standards and Technology, USA.

Huberman, B. and Adamic, L. (1999). Growth dynamics of the world wide web. *Nature*, 401:131.

ISO (2002). Multimedia content description interface part 4 (audio), 15938-4.

Itakura, F. (1975). Minimum prediction residual principle applied to speech recognition. *IEEE Trans. Acoust. Speech signal process.*, 23:67–72.

Itakura, F. (1976). Continuous speech recognition by statistical methods. *IEEE Proceedings*, 64:4:532–556.

Ito, A., Hoe, S.-P., Suzuki, M., and Makino, S. (2004). Comparison of features for dp-matching based query-by-humming system. In *Proceedings of the 5th International Conference on Music Information Retrieval*, Barcelona, Spain.

Iverson, P. and Krumhansl, C. L. (1993). Isolating the dynamic attributes of musical timbre. *Journal of the Acoustical Society of America*, 94:2595–2603.

Jehan, T. (2004). Perceptual segment clustering for music description and time-axis redundancy cancellation. In *Proceedings of the International Symposium on Music Information Retrieval (ISMIR). Barcelona, Spain*.

Jiang, D.-N., Lu, L., Zhang, H.-J., Tao, J.-H., and Cai, L.-H. (2002). Music type classification by spectral contrast feature. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME), Lausanne (Switzerland)*.

Jin, Q. and Waibel, A. (2000). A nave de-lambing method for speaker identification. In *Proceedings of International Conference on Spoken Language Processing 2000 (ICSLP 2000)*.

Johnston, J. (February 1988). Transform coding of audio signals using perceptual noise criteria. *IEEE Journal on Selected Areas in Communications*, 6(2):314–323.

Julesz, B. (1973). Experiments in the visual perception of texture. *Scientific American*, 232:34–43.

Julesz, B. (1981). Textons, the elements of texture perception, and their interactions. *Nature*, 290:91–97.

Julesz, B., Gilbert, E. N., Shepp, L. A., and Frisch, H. L. (1973). Inability of humans to discriminate between visual textures that agree in second-order statistics - revisited. *Perception*, 2:391–405.

Kapur, A., Benning, M., and Tzanetakis, G. (2004). Query-by-beat-boxing: Music retrieval for the dj. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR), Barcelona, Spain*.

Kendall, R. A. and Carterette, E. C. (1991). Perceptual scaling of simultaneous wind instrument timbres. *Music Perception*, 8:369–404.

Kim, Y. E. and Whitman, B. (2002). Singer identification in popular music recordings using voice coding features. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, Paris, France.

Kinnunen, T., Krkkinen, I., and Frnti, P. (2001). Is speech data clustered? - statistical analysis of cepstral features. In *European Conf. on Speech Communiation and Technology, (EUROSPEECH'2001), Aalborg, Denmark*, volume 4, pages 2627–2630.

Kohonen, T. (1995). *Self-Organizing Maps*. Springer-Verlag, Heidelberg.

Koolwaaij, J. W. and Boves, L. (1997). A new procedure for classifying speakers in speaker verification systems. In *Proceedings of EUROSPEECH '97, Rhodes.*

Krimphoff, J., McAdams, S., and Winsberg, S. (1994). Caractrisation du timbre des sons complexes. ii: Analyses acoustiques et quantification psychophysique. *Journal de Physique*, 4(5):625–628.

Kulesh, V., Sethi, I., and V., P. (2003). Indexing and retrieval of music via gaussian mixture models. In *Proceedings of the 3rd International Workshop on Content-Based Multimedia Indexing (CBMI), Rennes (France)*.

Kurzweil, R. (1996). *When Will HAL Understand What We Are Saying? Computer Speech Recognition and Understanding*, chapter 7, pages 131–170. The MIT Press.

Lee, J. H. and Downie, J. S. (2004). Survey of music information needs, uses and seeking behaviours: Preliminary findings. In *Proceedings of the Fifth International Conference on Music Information Retrieval, Barcelona, Spain*.

Lerski, R., Straughan, K., Shad, L., Boyce, D., Bluml, S., and Zuna, I. (1993). Mr image texture analysis an approach to tissue characterisation. *Magnetic Resonance Imaging*, 11:873–887.

Li, T. and Tzanetakis, G. (October 2003). Factors in automatic musical genre classification of audio signals. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA) New Paltz, NY (USA)*.

Lidy, T. and Rauber, A. (2005). Evaluation of feature extractors and psychoacoustic transformations for music genre classification. In *Proceedings of the 6th International Conference on Music Information Retrieval, London, UK*.

Lim, L., Wang, M., and Vitter, J. S. (2003). Sash: A self-adaptive histogram set for dynamically changing workloads. In *In Proceedings of 29th International Conference on Very Large Data Bases*.

Lin, N. and Zhao, H. (2005). Are scale-free networks robust to measurement errors? *Bioinformatics*, 6:119.

Liu, D., Lu, L., and Zhang, H.-J. (2003). Automatic mood detection from acoustic music data. In *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR), Baltimore, Maryland, USA*.

Logan, B. (October 2000). Mel frequency cepstral coefficients for music modeling. In *Proceedings of the 1st International Symposium of Music Information Retrieval (ISMIR), Plymouth, Massachusetts (USA)*.

Logan, B. and Salomon, A. (August 2001). A music similarity function based on signal analysis. In *in proceedings IEEE International Conference on Multimedia and Expo (ICME), Tokyo (Japan)*.

Maddage, N. C., Xu, C., and Wang, Y. (2003). A svm-based classification approach to musical audio. In *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR), Baltimore, Maryland (USA)*.

Martin, K. (1998). Towards automatic sound source recognition: Identifying musical instruments. In *Proc. NATO Computational Hearing Advanced Study Institute, Italy*.

Matsumoto, M. and Nishimura, T. (1998). Mersenne twister: A 623-dimensionally equidistributed uniform pseudorandom number generator. *ACM Trans. on Modeling and Computer Simulation*, 8(1):3–30.

McAdams, S., Winsberg, S., Donnadieu, S., De Soete, G., and Krimphoff, J. (1995). Perceptual scaling of synthesized musical timbres: Common dimensions, specificities, and latent subject classes. *Psychological Research*, 58:177–192.

McKay, C., Fiebrink, R., McEnnis, D., Li, B., and Fujinaga, I. (2005). Ace: A framework for optimizing music classification. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR), London, UK*.

McKay, C. and Fujinaga, I. (2004). Automatic genre classification using large high-level musical feature sets. In *Proceedings of International Symposium of Music Information Retrieval, Barcelona (Spain)*.

MIT (1995). Vistex database. `http://vismod.media.mit.edu/`.

Monk, E., Keller, J. P., Bohnenberger, K., and Daconta, M. C. (2000). *Java Pitfalls: Time-Saving Solutions and Workarounds to Improve Programs*. John Wiley & Sons. Article "When Runtime.exec() won't", retrieved on 16/01/2005 from `http://www.javaworld.com/javaworld/jw-12-2000/jw-1229-traps.html`.

Nabney, I. (2001). *Netlab: Algorithms for Pattern Recognition*. Springer (Advances in Pattern Recognition Series).

Nwe, T. L. and Wang, Y. (2004). Automatic detection of vocal segments in popular songs. In *Proceedings of International Symposium of Music Information Retrieval, Barcelona (Spain)*.

Pachet, F. (2003). Content management for electronic music distribution: The real issues. *Communications of the ACM*.

Pachet, F. and Cazaly, D. (2000). A taxonomy of musical genres. In *Proceedings RIAO*.

Pachet, F., LaBurthe, A., Zils, A., and Aucouturier, J.-J. (2004). Popular music access: The sony music browser. *Journal of the American Society for Information (JASIS), Special Issue on Music Information Retrieval*.

Pachet, F., Westermann, G., and Laigre, D. (2001). Musical datamining for emd. In *Proceedings Wedelmusic*.

Pachet, F. and Zils, A. (2003). Evolving automatically high-level music descriptors from acoustic signals. In *Springer Verlag LNCS, 2771*.

Pampalk, E. (2004). A matlab toolbox to compute music similarity from audio. In *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR), Barcelona, Spain*.

Pampalk, E., Dixon, S., and Widmer, G. (2003). On the evaluation of perceptual similarity measures for music. In *Proceedings of the Sixth International Conference on Digital Audio Effects DAFX, London (UK)*.

Peeters, G., LaBurthe, A., and Rodet, X. (2002). Toward automatic music audio summary generation from signal analysis. In *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR), Paris (France)*.

Peltonen, V., Eronen, A., Parviainen, M., and Klapuri, A. (2001). Recognition of everyday auditory scenes: Potentials, latencies and cues. In *Proceedings of the 110th Convention of the Audio Engineering Society, Amsterdam (The Netherlands)*.

Perrot, D. (1999). Scanning the dial : an exploration of factors in the identification of musical style. In *Proceedings of the 1999 Society for Music Perception and Cognition*.

Pestoni, F., Wolf, J., Habib, A., and Mueller, A. (2001). Karc: Radio research. In *Proc. WedelMusic Conference, Firenze, Italy*.

Platt, J. R. (1964). Strong inference: Certain systematic methods of scientific thinking may produce much more rapid progress than others. *Science*, 146(3642).

Plomp, R. (1976). *Aspects of Tone Sensation. A psychophysocal study.* London: Academic Press.

Plumbley, M. D., Abdallah, S. A., Bello, J. P., Davies, M. E., Monti, G., and Sandler, M. B. (2002). Automatic music transcription and audio source separation. *Cybernetics and Systems*, 33(6):603–627.

Press, W., Flannery, B., Teukolsky, S., and Vetterling, W. (1986). *Numerical Recipes, The Art of Scientific Computing*. Cambridge University Press, Cambridge.

Quinlan, J. (1993). *C4.5: Programs for Machine Learning*. Morgan Kauffman.

Rabiner, L. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2).

Rabiner, L. and Juang, B. (1993). *Fundamentals of speech recognition*. Prentice-Hall.

Ribowski, M. (1989). *He's A Rebel. The Truth About Phil Spector, Rock and Roll's Legendary Madman.* New York: E.P. Dutton.

Rosenberg, A. E., Delong, J., Lee, C.-H., Juang, B.-H., and Soong, F. K. (1992). The use of cohort normalized scores for speaker recognition. In *Proceedings of International Conference on Spoken Language Processing 1992, Banff*, pages 599–602.

Rossignol, S. (1998). Feature extraction and temporal segmentation of acoustic signals. In *Proceedings of the International Computer Music Conference*.

Roy, P., Aucouturier, J.-J., Pachet, F., and Beurivé, A. (2005). Exploiting the tradeoff between precision and cpu-time to speed up nearest neighbor search. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, London, UK.

Rubner, Y., Tomasi, C., and Guibas, L. (1998). The earth movers distance as a metric for image retrieval. Technical report, Stanford University.

Samet, H. (1989). *Applications of Spatial Data Structures*. Addison-Wesley.

Scaringella, N. and Zoia, G. (2005). On the modelling of time information for automatic genre recognition systems in audio signals. In *Proc. International Symposium on Music Information Retrieval, London (UK)*.

Scheirer, E. (1998). Tempo and beat analysis of acoustic musical signals. *Journal of the Acoustic Society of America*, 103(1):588–601.

Scheirer, E. and Slaney, M. (1997). Construction and evaluation of a robust multifeature speech/music discriminator. In *Proceedings of ICASSP*.

Schroeder, M. R., Atal, B. S., and Hall, J. L. (1979). Optimizing digital speech coders by exploiting masking properties of the human ear. *Journal of the Acoustical Society of America*, 66(6):16471652.

Singh, P. and Bregman, A. (1993). Effect of differences in temporal-amplitude envelope features and number of harmonics on perceptual segregation of tonal sequences. *Journal of the Acoustical Society of America*.

Slaney, M. (1998). Auditory toolbox v.2. Technical report 1998-010, retrieved on 16/01/2005 from `http://rvl4.ecn.purdue.edu/~malcolm/interval/1998-010/`, Interval Research Corporation.

Smith, J. O. and Abel, J. S. (1999). The bark and erb bilinear transforms. *IEEE Transactions on Speech and Audio Processing*.

Smith, L., McNab, R., and Witten, I. (1998). Sequence-based melodic comparison: A dynamic programming approach. *Computing in Musicology*, 11:101–117.

Soltau, H. (1998). Recognition of musical types. In *proceedings ICASSP*.

Stenzel, R. and Kamps, T. (2005). Improving content-based similarity masures by training a collaborative model. In *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR), London, UK*.

Talupur, M., Nath, S., and Yan, H. (2000). Classification of music genre. Working Paper, Computer Science Department Carnegie Mellon University.

Terasawa, H., Slaney, M., and Berger, J. (2005). The thirteen colors of timbre. In *Proceedings of the 2005 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, New Paltz, NY*.

Thaper, N., Guha, S., Indyk, P., and Koudas, N. (2002). Dynamic multidimensional histograms. In *Proceedings of the 2002 ACM SIGMOD international conference on Management of data, Madison, Wisconsin, USA*.

Tsai, W.-H. and Wang, H.-M. (2003). Towards automatic identification of singing language in popular music recordings. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, Barcelona, Spain.

Tuceryan, M. and Jain, A. K. (1998). Texture analysis. In Chen, C. H., Pau, L. F., and Wang, P. S. P., editors, *The Handbook of Pattern Recognition and Computer Vision (2nd Edition)*, pages 207–248. World Scientific Publishing Co.

Tukey, J. W., B. P. B. and Healy, M. J. R. (1963). The quefrency alanysis of time series for echoes: cepstrum, pseudo-autocovariance, cross-cepstrum, and saphe cracking. In *Proceedings of the Symposium on Time Series Analysis (M. Rosenblatt, Ed), Chapter 15, 209-243. New York: Wiley*.

Tversky, A. (1977). Features of similarity. *Psychol. Review*, 84(4):327–352.

Tzanetakis, G. and Cook, P. (1999). Multifeature audio segmentation for browsing and annotation. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, New Paltz, NY*.

Tzanetakis, G. and Cook, P. (2000). Audio information retrieval (air) tools. In *Proc. International Symposium on Music Information Retrieval*.

Tzanetakis, G. and Cook, P. (2002). Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5).

Tzanetakis, G., Essl, G., and Cook, P. (2001). Automatic musical genre classification of audio signals. In *proceedings ISMIR*.

Ulusoy, I. and Bishop, C. M. (2005). Generative versus discriminative models for object recognition. In *In Proceedings IEEE International Conference on Computer Vision and Pattern Recognition, CVPR., San Diego.*

Vignoli, F. and Pauws, S. (2005). A music retrieval system based on user driven similarity and its evaluation. In *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR), London, UK*.

Vincent, E. and Rodet, X. (2003). Instrument identification in solo and ensemble music using indepndnt subspace analysis. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, Barcelona, Spain.

Voorhes, E. and Harman, D. (1999). Overview of the eighth text retrieval conference. In *Proceedings of the Eighth Text Retrieval Conference (TREC), Gaithersburg, Maryland (USA)*. http://trec.nist.gov.

Welsh, M., Borisov, N., Hill, J., von Behren, R., and Woo, A. (1999). Querying large collections of music for similarity. Technical Report Technical Report UCB/CSD00 -1096, U.C. Berkeley Computer Science Division.

Wessel, D. L. (1991). Timbre space as a musical control structure. *Computer Music Journal*, 3(2):45–52.

West, K. and Cox, S. (2004). Features and classifiers for the automatic classification of musical audio signals. In *Proc. International Symposium on Music Information Retrieval, Barcelona (Spain)*.

Whitman, B. and Ellis, D. P. W. (2004). Automatic record reviews. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR), Barcelona (Spain)*.

Whitman, B. and Smaragdis, P. (2002). Combining musical and cultural features for intelligent style detection. In *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR), Paris (France)*.

Witten, I. H. and Frank, E. (2005). *Data Mining: Practical machine learning tools and techniques*. 2nd Edition, Morgan Kaufmann, San Francisco.

Yang, C. (2002). The macsis acoustic indexing framework for music retrieval: An experimental study. In *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR), Paris (France)*.

Young, S. J., Woodland, P. C., and Byrne, W. J. (1993). Htk: Hidden markov model toolkit v1.5. Technical report, Cambridge University Engineering Department Speech Group and Entropic Research Laboratories Inc.

Zils, A. and Pachet, F. (2004). Automatic extraction of music descriptors from acoustic signals using eds. In *Proceedings of the 116th AES Convention, Berlin*.