Transformées redondantes pour la représentation de signaux audio : application au codage et à l'indexation

Emmanuel Ravelli

THÈSE DE DOCTORAT
UNIVERSITÉ PARIS 6 - PIERRE ET MARIE CURIE
ECOLE DOCTORALE EDITE DE PARIS

Spécialité

Informatique, Télécommunications et Électronique

Présentée par

M. Emmanuel RAVELLI

Pour obtenir le grade de

Docteur de l'université Paris 6

Sujet de la thèse

Transformées redondantes pour la représentation de signaux audio : application au codage et à l'indexation

Thèse dirigée par

Laurent DAUDET - Université Paris 6

et

Gaël RICHARD - TELECOM ParisTech

Soutenance prévue le 27 Octobre 2008

devant le jury composé de:

Laurent DAUDET	Université Paris 6	Directeur de thèse
Gaël RICHARD	TELECOM ParisTech	Directeur de thèse
Pierre DUHAMEL	CNRS Paris	Rapporteur
Bastiaan KLEIJN	KTH Stockholm	Rapporteur
Mike DAVIES	Université d'Edinburgh	Examinateur
Jean-Gabriel GANASCIA	Université Paris 6	Examinateur
Pierrick PHILIPPE	France Télécom R&D	Examinateur

Transformées redondantes pour la représentation de signaux audio : application au codage et à l'indexation

Résumé Cette thèse étudie de nouvelles techniques de représentation du signal pour le codage audio. Les codeurs audio existants sont basés soit sur une transformée (codage par transformée), soit sur un modèle paramétrique (codage paramétrique), soit sur une combinaison des deux (codage hybride). D'une part, le codage par transformée permet une qualité transparente à haut débit (ex. AAC à 64 bkps/canal), mais obtient de mauvaises performances à bas débit. D'autre part, le codage paramétrique et le codage hybride obtiennent de meilleures performances que le codage par transformée à haut débit mais ne permettent pas une qualité transparente à haut débit. La nouvelle approche de représentation du signal que nous proposons permet d'obtenir une qualité transparente à haut débit et de meilleures performances que le codage par transformée à bas débit. Cette représentation du signal est basée sur un ensemble redondant de fonctions temps-fréquence composée d'une union de plusieurs bases MDCT à différentes échelles. La première contribution majeure de cette thèse est un algorithme à la fois rapide et performant qui décompose un signal dans cette ensemble redondant de fonctions. La deuxième contribution majeure de cette thèse est un ensemble de techniques qui permettent un codage de ces représentations à la fois performant et progressif. Finalement, cette thèse étudie l'application à l'indexation audio. Nous montrons que l'utilisation d'une union de plusieurs MDCT permet de dépasser les limitations des représentations utilisées dans les codeurs par transformée (en particulier la résolution fréquentielle), ce qui rend ainsi possible une indexation dans le domaine transformée performant.

Mots clés Traitement du signal, représentation des signaux, représentations parcimonieuses, transformées temps-fréquence, codage audio, quantification, indexation audio, classification.

Equipe d'acceuil Institut Jean le Rond d'Alembert, Equipe Lutherie Acoustique Musique (LAM), 11 rue Lourmel, 75015 Paris.

Audio signal representations with overcomplete transforms for coding and indexing

Abstract This thesis investigates new signal representations for audio coding. Existing state-of-the-art audio coders are based either on a transform (transform coding), or on a parametric model (parametric coding), or on a combination of both (hybrid coding). On the one hand, transform coding achieves (near-)transparent quality at high bitrates (e.g. AAC at 64kbps/channel), but gives poor performance at lower bitrates. On the other hand, parametric and hybrid coding achieve better performance than transform coding at low bitrates but cannot give transparent quality at high bitrates. The new approach for signal representation that we propose allows to achieve transparent quality at high bitrates, while giving better performance than transform coding at low bitrates. This signal representation is based on an overcomplete set of time-frequency functions composed by a union of several MDCT bases with different scales. The first major contribution of this thesis is a fast and efficient algorithm that decomposes a signal into this overcomplete set of functions. The second major contribution of this thesis is a set of techniques that allows the coding of these representations in an efficient and scalable way. Finally, this thesis investigates the application to audio indexing. We show that using a union of several MDCT bases allows to go beyond the limitations of the representations used in the transform coders (particularly the frequency resolution), which makes possible an efficient indexing in the transform domain.

Keywords Signal processing, signal representations, sparse representations, time-frequency transforms, audio coding, quantization, audio indexing, classification.

Résumé long

Nous proposons ici un long résumé en français de la thèse. Plutôt qu'un résumé exhaustif de la thèse, nous ne citerons que les points essentiels : une courte introduction comprenant les motivations ainsi que les contributions de la thèse; un bref état de l'art du codage audio et des représentations parcimonieuses; une étude de la représentation de signaux audio dans une union de bases MDCT; une étude de l'application au codage audio; une étude de l'application à l'indexation audio.

Introduction

Motivations

Le codage audio est né dans les années 80 de la nécessité de réduire le coût de stockage et de transmission des données audio numériques. Le premier format de codage audio standardisé est le MPEG-1, publié en 1993. La 3ème couche du standard MPEG-1 (MPEG-1 Layer 3 ou MP3) est la plus performante et elle connue un succès grandissant dans les années 90 avec le développement d'internet. Parallèlement à ce succès, la recherche en codage audio donna naissance à un nouveau standard, le codage audio avancé (Advanced Audio Coding ou AAC), publié en 1997 dans MPEG-2 et mis à jour en 1999 dans MPEG-4. Le AAC est plus performant que le MP3 et est considéré encore actuellement comme l'état de l'art en codage audio haute qualité. Contrairement au MP3, le AAC ne connu un succès que plus tard, avec le choix d'Apple de donner la priorité au format AAC pour ses produits. Bien que le codage audio haute qualité a atteint probablement une limite de performance avec le AAC, les performances à bas débit sont encore très mauvaises. La recherche en codage audio dans les années 2000 a ainsi donné naissance à plusieurs standards de codage audio bas débit, comme le MPEG-4 HE-AAC, le MPEG-4 SSC, et le 3GPP AMR-WB+. Ces nouveaux standards ne viennent pas en remplacement du AAC mais plutôt en complément, car ils sont performants à bas débit mais ne permettent pas de donner la même qualité transparente que le AAC à haut débit.

Il existe maintenant un grand nombre de codecs audio standardisés et performants. Mais chaque codec est spécialisé , par exemple pour un signal particulier comme la parole, ou encore pour une certaine plage de débit. Un des défis en codage audio actuellement est alors de concevoir un codec audio universelle, qui combine les avantages des codecs audio actuelles d'une manière adaptative et flexible. Un autre défi en codage audio est de combiner codage audio et indexation audio, par exemple en concevant une technique de représentation du signal utile à la fois pour le codage audio et l'indexation audio. Cela permettrait de concevoir des systèmes d'indexation audio rapides et performants qui fonctionnent à partir de fichiers codés.

Contributions

Contributions en représentation du signal Nous proposons dans cette thèse de nouvelles approches pour la représentation du signal qui sont utiles à la fois pour le codage audio et l'indexation audio. Ces nouvelles méthodes sont basés sur un domaine de recherche appelé "représentations parcimonieuses de signaux" et dont le but est de modéliser un signal comme la somme d'un petit nombre de fonctions élémentaires qui sont choisis parmi une collection (appelé dictionnaire) de taille arbitraire (généralement bien plus grande que la dimension du signal). Contrairement aux approches traditionnels du codage audio haute qualité où un signal est représenté grâce à une transformée temps-fréquence, la méthode proposée utilise une union redondante de plusieurs transformées à différents compromis temps-fréquence. Cela permet la modélisation d'un signal avec moins de fonctions élémentaires, une propriété qui a un intérêt évident en codage audio. Le principal problème d'une telle approche est de trouver un moyen efficace pour décomposer un signal dans cette ensemble redondant de fonctions. Nous proposons alors plusieurs algorithmes basés sur l'algorithme de Matching Pursuit: une implémentation rapide, un contrôle du pre-echo, et une optimisation débit-distortion. En plus de l'intérêt en codage audio, ce travail sur la représentation du signal a permis aussi d'étudier le problème fondamental en représentations parcimonieuses de signaux qui est le choix du dictionnaire.

Contributions en codage audio Nous proposons dans cette thèse deux codeurs audio basés sur les nouvelles approches de représentation du signal introduites précédemment. Le principal problème est de trouver des techniques de codage efficaces et flexibles qui sont adaptés aux représentations du signal proposés. Nous proposons alors plusieurs techniques qui permettent un codage audio progressif et performant: un algorithme d'entrelacement de coefficients, un algorithme de codage psychoacoustique en plan de bits, et un algorithme en plan de bits adaptatif. Le premier codeur audio est basé sur l'algorithme de codage psychoacoustique et a été évalué à la fois avec une mesure objective et un test d'écoute. Le deuxième codeur utilise l'approche adaptative et nous proposons dans ce cas uniquement des résultats préliminaires car une évaluation complète n'a pas encore été faite. Nous proposons aussi dans cette thèse une étude de la quantification polaire imbriquée, une technique qui peut être utilisée pour par ex. le codage audio progressif basé sur une transformée complexe comme la MCLT.

Contributions en indexation audio dans le domaine transformée Nous proposons dans cette thèse de nouvelles approches pour l'indexation audio dans le domaine transformée. Nous considérons à la fois de nouveaux codeurs audio et de nouvelles applications d'indexation. L'état de l'art dans ce domaine est assez limité: peu de résultats ont été publiés, essentiellement pour le codeur audio MP3. Notre travail considère non seulement le MP3, mais aussi le plus récent AAC ainsi qu'un des codeurs développé dans cette thèse. Pour chaque codeur, nous proposons des algorithmes simples et rapides qui calculent des descripteurs pour l'indexation audio, comme une fonction de détection, un chromagramme, et des coefficients MFCC. Nous étudions les performances de ces descripteurs ainsi que le temps de calcul pour 3 applications: le suivi de rythme, la reconnaissance d'accords, et la classification de genre musical.

Codage audio avec pertes: une introduction

Le principe du codage audio avec pertes est de réduire la redondance statistique et la non-pertinence perceptive d'un signal audio. Pour réaliser une telle tâche, un codeur utilise deux opérations principales (voir Fig. 0.1). Premièrement, le signal original est analysé à l'aide d'une technique de représentation du signal. Deuxièmement, les coefficients ou paramètres qui décrivent cette représentation du signal sont convertis en un format binaire à l'aide d'une technique de

codage de source. Selon les techniques utilisées, chaque étage du codeur audio réduit la redondance statistique et/ou la non-pertinence perceptive d'un signal audio. Le décodeur audio à la même configuration à deux étages, il convertit la séquence de bits produite par le codeur audio en un ensemble de coefficients ou paramètres puis synthétise le signal audio décodé à l'aide de ces coefficients ou paramètres. Nous proposons dans la suite une introduction des techniques les plus utilisées pour la représentation du signal et le codage de source, ainsi que des méthodes pour l'évaluation des codeurs audio avec pertes.

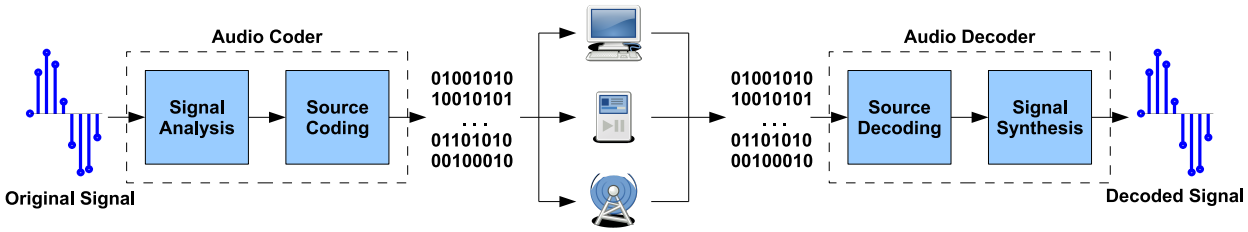

Figure 0.1: *Schéma bloc simplifié d'un codec audio.*

Représentation du signal

Il y a trois catégories principales de représentation du signal utilisé en codage audio avec pertes: les bancs de filtres, les modèles paramétriques, et les approches hybrides. Le principe du banc de filtres est de convertir un signal temporel en une représentation temps-fréquence composé de plusieurs signaux en sous-bandes. Les approches par banc de filtres sont utilisés dans les codeurs audio haute-qualité comme le MP3 ou le AAC. Contrairement aux bancs de filtres, les modèles paramétriques modélisent, en général, un sous-espace du signal original à l'aide d'un petit nombre de paramètres. Les modèles paramétriques sont utilisés dans les codeurs bas-débit pour la parole ou l'audio comme le AMR ou le SSC. Finalement, les approches hybrides combinent des techniques basées sur des bancs de filtres et des modèles paramétriques pour améliorer la performance. De telles approches sont utilisées dans des codeurs audio récents comme le HE-AAC ou le AMR-WB+.

Nous ne détaillerons pas ici toutes les techniques utilisées en codage audio, nous ne mentionnerons que la transformée en cosinus discrète modifiée (MDCT) car c'est le banc de filtres que nous étudierons principalement dans le reste de cette thèse. Considérons un vecteur (le signal audio) $\mathbf{x}$ de longueur $N = PK$ échantillons, composé de P segments de longueur K échantillons chacun. La matrice transformée $\mathbf{T}$ de taille $N \times N$ qui correspond à une MDCT avec une fenêtre de taille fixe $L = 2K$ est définie comme

$$\mathbf{T}\left(n, pK + k\right) = g_{p,k}(n) \ , \ 0 \leq p < P \ , \ 0 \leq k < K \ , \ 0 \leq n < N$$

avec

$$g_{p,k}(n) = w_p(u)\sqrt{\frac{2}{K}}\cos\left[\frac{\pi}{K}\left(u + \frac{K}{2} + \frac{1}{2}\right)\left(k + \frac{1}{2}\right)\right]$$

et

$$u = n - \left(p - \frac{1}{2}\right)K$$

et $w_p(u)$ est une fenêtre définie sur $0 \leq u < L$ et qui vérifie

$$\begin{aligned} w_0(u) &= 1 \ , \ 0 \leq u < L/2 \\ w_p^2(u + L/2) + w_{p+1}^2(u) &= 1 \ , \ 0 \leq u < L/2 \ , \ 0 \leq p < P - 1 \\ w_{P-1}(u + L/2) &= 1 \ , \ 0 \leq u < L/2. \end{aligned}$$

Les deux fenêtres les plus utilisées sont la fenêtre sinusoïdale et la fenêtre Kaisel-Bessel Derived (KBD). Il est important de noter que les codeurs de l'état de l'art comme le AAC utilisent une MDCT adaptative avec deux tailles de fenêtres et des fenêtres de transition entre les fenêtres longues et les fenêtres courtes.

Codage de source

Les techniques de représentation du signal introduites précédemment produisent un ensemble de coefficients ou paramètres, comme par ex. les coefficients d'une MDCT. Le but du codage source est alors de convertir ces coefficients ou paramètres en une séquence de bits et inversement, convertir cette séquence de bits en un ensemble de coefficients ou paramètres décodés. Le codage de source fait référence à deux opérations: le codage et le décodage. Le codeur prend en entrée une séquence de vecteurs à valeurs réelles et produit en sortie une séquence de bits à longueur variable. Dans les applications pratiques, cette séquence de bits est alors ajouté à un fichier ou transmise dans un réseau. Le décodeur prend en entrée la séquence de bits produite par le codeur et produit en sortie une séquence de vecteurs à valeurs réelles. Le processus de codage/décodage introduit nécessairement des dégradations: la sortie du décodeur n'est pas parfaitement égale à l'entrée du codeur. Cette perte est due au premier étage du codeur, appelé quantification, qui convertit la séquence de vecteurs à valeurs réelles en une séquence d'entiers. En codage audio, le système est conçu pour que cette perte ne soit pas ou peu perçu par le système auditif humain. Le deuxième étage du codeur est appelé codage entropique et convertit simplement la séquence d'entiers en une séquence de bits, de manière à minimiser le nombre de bits produits. Le décodeur a la même configuration à deux étages.

Il existe un grand nombre de techniques de codage de source. L'approche que nous utiliserons principalement dans le reste de cette thèse est appelée codage en plan de bits. Cette technique est une combinaison de quantification imbriquée et de codage entropique et permet un codage progressif.

Evaluation du codage audio avec pertes

Considérons un signal audio $\mathbf{x}$ appelé "signal de référence", un codeur audio avec pertes produit un autre signal $\hat{\mathbf{x}}$ appelé "signal test". Le but du codage audio est d'obtenir un signal de test avec un minimum de dégradations dans la qualité perçue et comparé au signal de référence. Il est ainsi nécessaire en codage audio d'évaluer ces dégradations d'un point de vue perceptif. Comme les auditeurs humains sont les juges ultimes de la qualité du signal de test, les tests d'écoutes formels sont considérés comme le seul moyen satisfaisant pour évaluer les performances d'un codeur audio. Cependant, les tests d'écoutes sont très long à réaliser car ils nécessitent l'implication d'un grand nombre de sujets. Ceci est particulièrement vrai lorsqu'on veut comparer un grand nombre de codeurs et de signaux. Par conséquent, il existe maintenant un certain nombre de mesures objectives de qualité audio qui sont censées être bien corrélées avec les résultats des tests d'écoutes. Bien que ces mesures sont loin d'être parfaites et ne peuvent pas remplacer les tests d'écoutes, elles donnent une bonne indication générale de la performance d'un codeur et sont utiles par exemple pour régler les paramètres d'un codeur.

Nous utiliserons dans cette thèse: un test d'écoute MUSHRA, destiné à évaluer les codeurs de qualité intermédiaire (généralement bas débit); et la mesure objective PEMO-Q, basé sur un modèle de l'audition développé à l'université d'Oldenburg.

Représentations parcimonieuses de signaux

Les représentations parcimonieuses de signaux cherchent à modéliser un signal avec un petit nombre de fonctions élémentaires. Ces fonctions sont choisies parmi une collection de taille arbitraire, qui peut être bien plus grande que la taille du signal. Les représentations parcimonieuses de signaux trouvent des applications évidentes en codage, que ce soit en codage audio, codage d'image ou codage vidéo. Mais elles trouvent aussi un succès grandissant dans d'autres applications, comme la séparation de sources, le débruitage, la reconnaissance de formes...

Problèmes et solutions

Notations mathématiques Considérons un vecteur colonne (le signal) $\mathbf{x}$ dans $\mathcal{R}^N$. Les représentations parcimonieuses consistent à approximer le signal $\mathbf{x}$ (avec une possible égalité) par une combinaison linéaire de K vecteur colonnes unitaires $\mathbf{g}_k$

$$\mathbf{x} \simeq \sum_{k=1}^{K} c_k \mathbf{g}_k$$

Les vecteurs $\mathbf{g}_k$ appartiennent à $\mathcal{R}^N$ et sont appelés "atomes", les scalaires c_k sont les coefficients. La collection finie de tous les atomes est appelée le "dictionnaire" et s'écrit

$$\mathcal{D} = \{\mathbf{g}_k, 1 \leq k \leq K\}$$

Pour simplifier les notations, nous définissons la matrice synthèse $\mathbf{\Phi}$ de taille $N \times K$, où les colonnes $\mathbf{\Phi}_k$ de $\mathbf{\Phi}$ correspondent aux atomes $\mathbf{g}_k$; $\mathbf{\Phi}_\Gamma$; la sous-matrice de $\mathbf{\Phi}$ qui contient seulement les colonnes de $\mathbf{\Phi}$ aux indices dans Γ; le vecteur colonne $\mathbf{c}$, où les éléments sont les coefficients c_k; et $\mathbf{c}_\Gamma$ est le sous-vecteur de $\mathbf{c}$ qui contient seulement les éléments de $\mathbf{c}$ dont les indices sont dans Γ. On a alors

$$\mathbf{x} \simeq \mathbf{\Phi c}$$

S'il y a égalité dans l'équation précédente, alors le signal est représenté exactement par une combinaison linéaire d'atomes, et le vecteur de coefficients $\mathbf{c}$ est alors appelé "représentation exacte" (ou simplement "représentation"). Sinon, le signal est approximé avec plus ou moins de précision par une combinaison linéaire d'atomes, le vecteur $\mathbf{c}$ est alors appelé "approximation", et $\hat{\mathbf{x}} = \mathbf{\Phi c}$ est appelé "approximant" de $\mathbf{x}$. On considère ces deux cas dans la suite.

Représentations exactes On peut distinguer ici les deux cas les plus rencontrés. Si le dictionaire est une base orthonormale, alors la représentation est unique et est donné par $\mathbf{c} = \mathbf{\Phi}^T \mathbf{x}$. Si le dictionaire est complet et redondant, alors la représentation n'est pas unique et le but des représentations parcimonieuses est alors de trouver "la représentation la plus parcimonieuse", ou autrement dit celle dont l'énergie est concentrée sur le plus petit nombre de coefficients. Plusieurs critères de "parcimonie" existent, le plus utilisé étant la "quasi-norme p" définie comme $E^{(p)}(\mathbf{c}) = \left(\sum_{k=1}^{K} |c_k|^p\right)^{\frac{1}{p}}$ pour $p > 0$ et $E^{(0)}(\mathbf{c}) = \lim_{p \to 0} \sum_{k=1}^{K} |c_k|^p$. Trouver une représentation parcimonieuse revient alors à résoudre un problème d'optimisation où le critère de parcimonie est minimisée. Deux exemples connus sont "Basis Pursuit" et "FOCUSS".

Approximations Dans le cas d'une approximation du signal, les représentations parcimonieuses cherchent non seulement une approximation "parcimonieuse" mais aussi une approximation aussi proche que possible du signal. Il faut donc définir dans ce cas-là, en plus d'un critère de parcimonie, une mesure de l'erreur d'approximation. La plus simple et la plus utilisée est la norme l_2 mais d'autres existent en audio, en particulier les mesures de distortion psychoacoustique. Trouver une approximation parcimonieuse revient alors à résoudre un problème d'optimisation, soit en contraignant la mesure de parcimonie, soit en contraignant l'erreur d'approximation, soit en utilisant une méthode lagrangienne. Des exemples connus sont "Basis Pursuit denoising", les algorithmes de seuil, "LASSO", "Regularized FOCUSS". Le principal inconvénient de ces approches est leur complexité calculatoire. Une alternative moins coûteuse est d'utiliser des algorithmes itératifs qui sélectionnent un atome à chaque itération. Ces algorithmes sont appelés "algorithmes gloutons" et sont détaillés dans la suite.

Algorithmes gloutons

Les algorithmes gloutons sont des algorithmes itératifs qui sélectionnent à chaque itération un atome. L'algorithme le plus simple et le plus connue est l'algorithme "Matching Pursuit". A chaque itération, il sélectionne l'atome le plus corrélé avec le résidu, puis le soustrait au résidu. L'algorithme s'arrête lorsqu'un critère d'arrêt est vérifié, comme par exemple un seuil sur le rapport signal à bruit. L'algorithme Matching Pursuit est détaillé ci-dessous.

Algorithm 1: Algorithme "Matching Pursuit"

Input: Le signal $\mathbf{x}$ et le dictionaire Φ

Output: Le vecteur de coefficients $\mathbf{c}$ et le résidu $\mathbf{r}$ telle que $\mathbf{x} = \Phi\mathbf{c} + \mathbf{r}$

1 Initialisation: $\mathbf{r} = \mathbf{x}$, $\mathbf{c} = \mathbf{0}$;

2 **repeat**

3 Produits scalaires: $\mathbf{a} = \Phi^T\mathbf{r}$;

4 Trouver le maximum: $k_{max} = \underset{1 \le k \le K}{\operatorname{argmax}} |a_k|$;

5 Mis à jour des coefficients: $c_{k_{max}} = c_{k_{max}} + a_{k_{max}}$;

6 Mis à jour du résidu: $\mathbf{r} = \mathbf{r} - a_{k_{max}}\Phi_{k_{max}}$;

7 **until** *un critère d'arrêt est satisfait* ;

L'algorithme Matching Pursuit est le plus simple et le plus répandu, mais il existe d'autres algorithmes gloutons, comme par exemple l'algorithme Matching Pursuit Orthogonal (OMP) qui est plus performant que Matching Pursuit mais aussi plus complexe. Une alternative récente à OMP sont les algorithmes de "Gradient Pursuit", qui permettent une performance proche de OMP et une complexité proche de Matching Pursuit.

Dictionnaires

Un des problèmes des représentation parcimonieuses est la conception du dictionnaire. Une première approche consiste à concevoir les formes d'ondes des atomes à partir d'une base de signaux à modéliser et en utilisant des méthodes d'apprentissage. Une deuxième méthode, celle qui nous intéresse, consiste à utiliser des formes d'ondes déterministes. Nous donnons quelques exemple ci-dessous de dictionnaires utilisés dans les applications de codage audio/image/vidéo.

Dictionnaires orthogonaux Le dictionnaire orthogonal le plus simple est le dictionnaire de dirac, cela correspond à un signal PCM. En audio, on obtient des approximations plus parci-

monieuses avec des dictionnaires d'atomes temps-fréquence, comme la MDCT, le plus utilisée en codage audio et déjà introduite précédemment. En codage d'image, les dictionnaires orthogonaux les plus utilisés sont la DCT (utilisé dans le codeur JPEG), et la DWT (utilisée dans le codec JPEG 2000). La DCT est aussi utilisée en codage vidéo.

Dictionnaires redondants Un dictionnaire redondant très utilisé en audio est le dictionnaire de Gabor, un ensemble d'atomes sinusoïdaux fenêtrés à échelle et fréquence variable. Un autre exemple d'atome utilisé en audio est "l'atome harmonique" composé de la somme de plusieurs atomes de Gabor dont leurs fréquences sont reliées harmoniquement. Pour l'image, les atomes de Gabor à deux dimensions sont utilisés par exemple en codage de vidéo pour coder le résidu de prédiction. Un autre dictionnaire redondant utilisé en image est composé d'atomes générés par l'application de transformations géométriques (translation, rotation, étirement) à une ou plusieurs fonctions génératrices. Ce type de dictionnaire est utilisé en codage d'image bas débit et en codage vidéo.

Représentation de signaux dans une union de bases MDCT

L'approche traditionnel en codage audio haute-qualité est d'utiliser la transformée MDCT (introduite précédemment), ou la variante adaptative de la MDCT qui utilise deux tailles de fenêtre. Nous proposons dans la suite une nouvelle représentation du signal pour le codage audio, basée sur une union de plusieurs MDCT à différentes échelles. Contrairement à l'approche traditionnel, cette nouvelle approche permet une approximation plus parcimonieuse du signal, en modélisant les parties stationnaires du signal avec un petit nombre d'atomes à grande échelle, et en modélisant les parties transitoires avec un petit nombre d'atomes à petite échelle.

Modèle du signal

Considérons un signal $\mathbf{x}$ de longueur N échantillons, il est approximé comme

$$\mathbf{x} = \mathbf{\Phi c} + \mathbf{r}$$

avec $\mathbf{r}$ le résidu, $\mathbf{c}$ le vecteur de coefficients de longeur MN et $\mathbf{\Phi}$ la matrice synthèse de taille $N \times MN$ défini comme la concaténation de M matrices T_m de taille $N \times N$, chaque matrice T_m correspondant à une MDCT avec une taille de fenêtre $L_m = L_0 2^m$. La matrice synthèse $\mathbf{\Phi}$ est alors définie comme

$$\mathbf{\Phi}\left(n, mN + pK_m + k\right) = g_{m,p,k}(n)\ ,\ 0 \leq m < M\ ,\ 0 \leq p < P_m\ ,\ 0 \leq k < K_m\ ,\ 0 \leq n < N$$

où P_m est tel que $N = P_m K_m$ et correspond au nombre de segments de longueur $K_m = L_m/2$, et

$$g_{m,p,k}(n) = w_{m,p}(u)\sqrt{\frac{2}{K_m}}\cos\left[\frac{\pi}{K_m}\left(u + \frac{K_m}{2} + \frac{1}{2}\right)\left(k + \frac{1}{2}\right)\right]$$

et

$$u = n - \left(p - \frac{1}{2}\right)K_m$$

La fenêtre $w_{m,p}(u)$ est définie sur $0 \leq u < L_m$ et est dans notre cas une fenêtre sinusoidale.

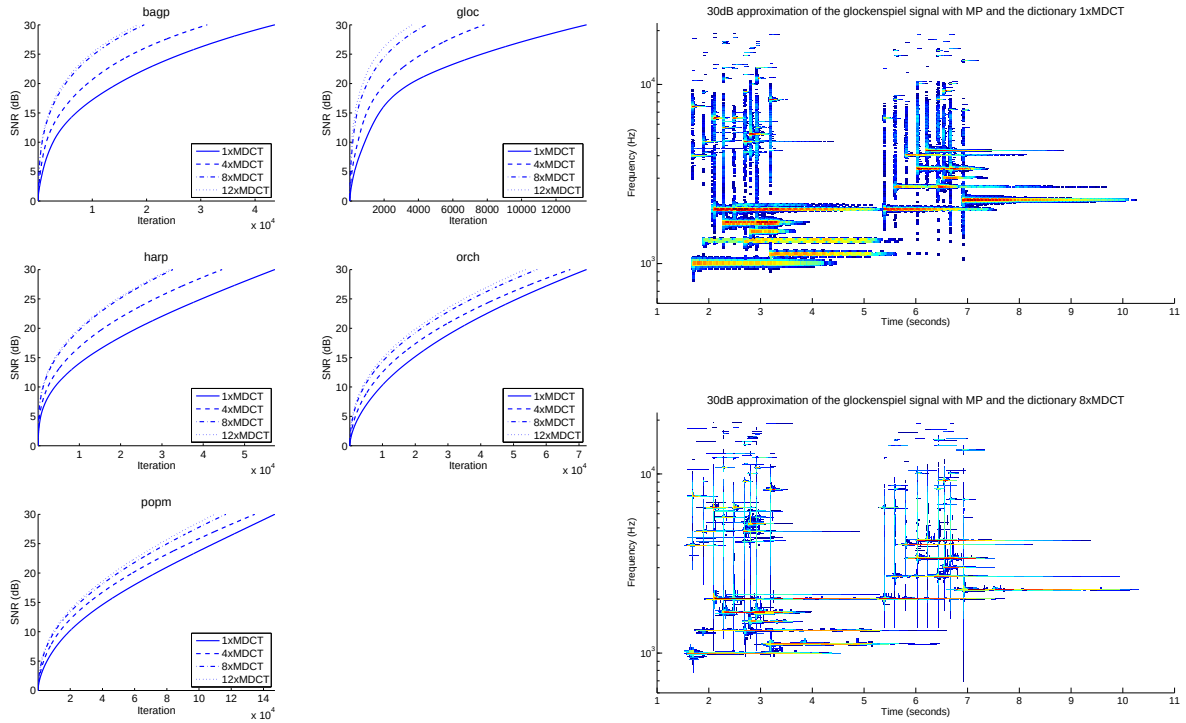

Figure 0.2: *Union de bases MDCT: performance et tracé temps-fréquence.*

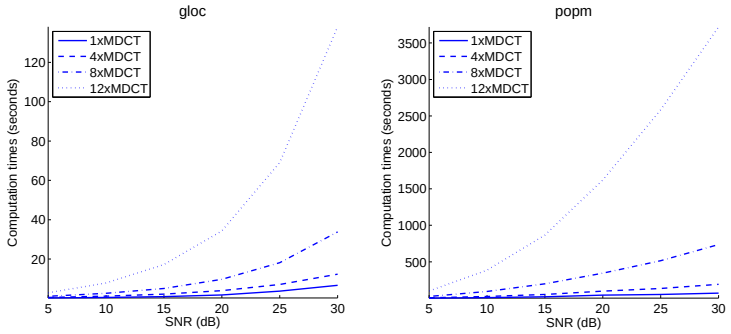

Figure 0.3: *Union de bases MDCT: temps de calcul.*

Expérience

Nous comparons dans la suite les performances obtenues avec plusieurs dictionnaires sur 5 signaux. Les dictionnaires sont: une MDCT à 2048 (1xMDCT), 4 MDCT de 512 à 4096 (4xMDCT), 8 MDCT de 128 à 16384 (8xMDCT), et 12 MDCT de 32 à 65536 (12xMDCT). Les signaux sont approximés dans ces dictionnaires avec une implémentation rapide de l'algorithme Matching Pursuit: the Matching Pursuit ToolKit (MPTK). La Fig. 0.2 montre le SNR en fonction de l'itération (environ égale au nombre d'atomes) pour les 5 signaux et les 4 dictionnaires, ainsi qu'un tracé temps-fréquence pour un signal et deux dictionaires. Ces résultats montrent premièrement qu'augmenter le nombre de MDCT dans le dictionnaire augmente la parcimonie du signal. Cependant, le dictionnaire 12xMDCT obtient des performances très proche du dictionnaire 8xMDCT, il apparaît donc

que les performances sont bornées. Deuxièmement, ces résultats montrent que les performances sont très dépendants du signal, l'approximation du signal gloc est par exemple bien plus parcimonieuse que celle du signal orch. Le tracé temps-fréquence du signal gloc montre bien l'excellente performance obtenue avec ce signal. Contrairement au dictionnaire 1xMDCT, le dictionnaire 8xMDCT permet de modéliser les parties stationnaires du signal avec un petit nombre d'atomes à grande échelle, et de modéliser les parties transitoires avec un petit nombre d'atomes à petite échelle. La Fig. 0.3 montre les temps de calcul obtenus avec 2 signaux et les 4 dictionaires. Ces résultats montrent que le temps de calcul augmente grandement avec la taille du dictionaire. Cela est dû aux MDCT de grande taille de fenêtre.

Comme le dictionaire 8xMDCT obtient des performances proches du dictionnaire 12xMDCT mais avec un temps de calcul grandement inférieure, nous choisirons pour la suite de cette thèse le dictionnaire 8xMDCT pour approximer un signal audio.

Algorithme Matching Pursuit modifié avec contrôle du pré-echo

L'algorithme de Matching Pursuit standard permet d'obtenir une représentation parcimonieuse comme nous l'avons vu précédemment. Cependant, nous avons remarqué que cet algorithme aussi produit un artefact de pré-echo sur des signaux contenant de fortes attaques, comme par exemple le signal gloc. Nous proposons alors un algorithme de Matching Pursuit modifié, qui va tester à chaque itération si l'atome sélectionné introduit du pré-echo ou non. S'il introduit du pré-echo, il est enlevé du dictionnaire et un autre atome est selectionné. Pour savoir si un atome introduit du pré-echo, on calcule la corrélation entre l'atome et le signal dans des sous-fenêtres. S'il y a une grande variation entre ces corrélations (le ratio entre le max et le min est plus grand qu'un seuil) alors l'atome introduit du pré-echo. Pour régler le paramètre de seuil, nous avons réaliser un certain nombre d'expérience, en particulier à l'aide d'une mesure proposé par B. Sturm appelé "Dark Energy" et qui mesure l'interférence entre les atomes successifs sélectionnés par l'algorithme de Matching Pursuit. Les résultats ont montrés une préférence pour une valeur de seuil de 100.

Union de bases MDCT pour le codage audio

Nous décrivons dans la suite un codeur audio progressif basé sur la nouvelle approche de représentation de signaux introduite précédemment.

Groupement et entrelacement

D'une part, l'approximation du signal $\mathbf{c}$ est calculé par l'algorithme Matching Pursuit pour le signal entier. D'autre part, pour améliorer l'efficacité de codage, on utilise une approche de codage en trame-par-trame. Il est donc nécessaire de "découper" le vecteur de coefficient $\mathbf{c}$ en trames. On utilise pour cela une approche en deux étapes. L'ensemble des coefficients est d'abord partitionné en sous-ensemble de coefficients correspondant à des segments temporels de taille fixe est égale à la moitié de la plus grande taille de fenêtre. Ces sous-ensembles sont appelés "slot", sont définis comme $\mathcal{S}_q = \left\{ c_{m,p,k} \mid \text{floor}\left(\frac{p-(P'_m-1)}{P'_m}\right) = q \right\}$ avec $P'_m - 1 = 2^{M-m-1} - 1$, et sont illustrés Fig. 0.4. La deuxième étape consiste à créer un vecteur de coefficients par "slot" grâce à un algorithme d'entrelacement. Cet algorithme d'entrelacement est illustré Fig. 0.4 pour un cas simple ($M = 3$ et $L_0 = 2$).

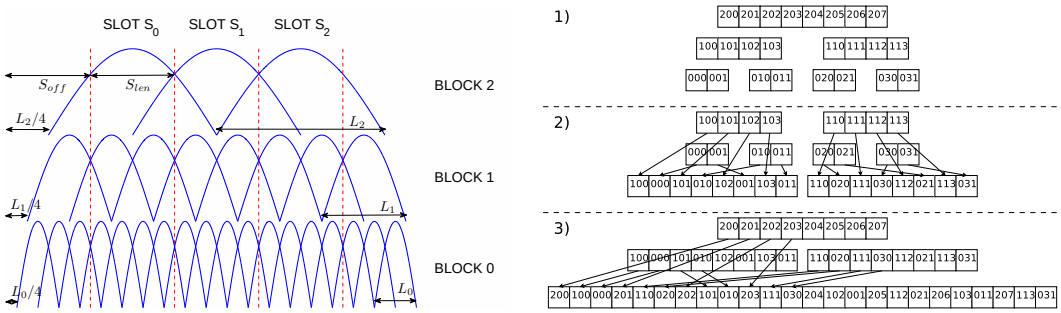

Figure 0.4: *Groupement et entrelacement: examples simple où* $M = 3$.

Codage en plan de bits

Chaque vecteur de coefficients entrelacés est ensuite codé avec un algorithme de codage en plan de bits. Cet algorithme permet un codage progressif de l'information. Le principe basique est le suivant: les coefficients sont d'abord normalisés par le maximum des amplitudes des coefficients; ce maximum est quantifié et codé; les coefficients sont alors représentés sous une forme signe/amplitude où l'amplitude est quantifié de manière très fine et représenté sous une forme binaire; l'algorithme code ensuite successivement chaque plan de bits (un plan de bits étant le vecteur des bits de même poids) partant des bits de poids fort; un plan de bits est codé en deux étapes, une étape de signifiance et une étape de raffinement; l'étape de signifiance code la position des coefficients pour lesquels un "1" apparaît pour la première fois, l'étape de rafinement code les bits de rafinements pour les coefficients dont la position à déjà été codé; on utilise un algorithme basé sur des codes de Golomb adaptatifs pour l'étape de signifiance. L'algorithme en plan de bits est détaillé ci-dessous

Algorithm 2: Codage en plan de bits

Input: Un vecteur de coefficients entrelacés $\mathbf{v} = \{v_i | i = 1...ML_{M-1}\}$
Output: La séquence de bits de sortie
Quantifier et coder l'amplitude maximale $A = max(abs(v_i))$;
Initialisation: $\mathbf{z} = \mathbf{0}$, $\nu = 1$;
repeat
  Calculer un plan de bits: $b_i = mod(floor(abs(v_i) * 2^{\nu}/A), 2)$ for all i;
  Etape de signifiance: code $BS = \{b_i | z_i = 0\}$ and signs;
  Etape de rafinement: code $BR = \{b_i | z_i = 1\}$;
  Mis à jour de la signifiance: $z_i = 1$ for all i such that $b_i \in BS$ and $b_i = 1$;
  Iteration: $\nu = \nu + 1$;
until *le budget de bits dépensé ou* $\nu > \nu_{max}$;

Nous proposons ensuite un algorithme de codage en plan de bits modifié qui prend en compte des considérations psychoacoustiques et permet une organisation de l'information où les composantes les plus importantes perceptivement sont codées en premier. Pour cela, on utilise une généralisation du modèle de Johnston à une représentation de signaux multi-échelles. Ce modèle psychoacoustique est calculé à partir d'un signal synthétisé en utilisant les coefficients partiellement codés. Cela permet de ne pas transmettre les paramètres du modèle et ainsi réduire le coût de codage car le modèle est calculé de la même manière sur les coefficients partiellement décodés au niveau du décodeur. Pour intégrer ce modèle dans l'algorithme de codage en plan de bits, on utilise une

méthode simple qui consiste à coder un sous-ensemble de chaque plan de bits à chaque itération de l'algorithme, ce sous-ensemble correspond aux coefficients dont le niveau de quantification est le plus loin du niveau de masque donné par le modèle, cela revient à coder les coefficients qui minimise le rapport bruit à masque. Cet algorithme est détaillé ci-dessous.

Algorithm 3: Codage en plan de bits psychoacoustique

Input: Un vecteur de coefficients entrelacés $\mathbf{v} = \{v_i | i = 1...ML_{M-1}\}$
Output: La séquence de bits de sortie
Quantifier et coder l'amplitude maximale $A = max(abs(v_i))$;
Initialisation: $\mathbf{z} = \mathbf{0}$, $\nu = \mathbf{1}$;
Initialiser le masque psycho. **th** to the ATH;
repeat
  if *U nouveau bits on été ajoutés à la séquence de bits* **then**
    mettre à jour le masque **th**;
  end
  Calculer la différence: $g_i = -log2(\sqrt{th_i}/A) - \nu_i$ for all i;
  Selectionner: $s_i = g_i \geq mean(g_i)$ for all i;
  Calculer le plan de bits: $b_i = mod(floor(abs(v_i) * 2^{\nu_i}/A), 2)$ for all i;
  Etape de signifiance: code $BS = \{b_i | z_i = 0 \text{ and } s_i = 1\}$ and signs;
  Etape de rafinement: code $BR = \{b_i | z_i = 1 \text{ and } s_i = 1\}$;
  Mettre à jour la signifiance: $z_i = 1$ for all i such that $b_i \in BS$ and $b_i = 1$;
  Iteration: $\nu = \nu + 1$;
until *le budget de bits dépensé ou* $\nu > \nu_{max}$;

Résultats

Evaluation objective Nous proposons dans un premier temps une évaluation objective du codeur audio proposé. Pour cela, nous utilisons le logiciel PEMO-Q qui produit deux mesures: la mesure "Perceptual Similarity Measure (PSM)" et la mesure "Objective Difference Grade". La Fig. 0.5 montre les résultats de deux expériences. La première compare le codeur proposé avec un codeur par transformée de référence. Le codeur par transformée utilise exactement le même système de codage, seul la représentation du signal est différente: c'est une MDCT simple avec une taille de fenêtre égale à 2048 échantillons. Cette première expérience montre la supériorité de la nouvelle approche de représentations de signaux. La deuxième expérience compare notre codeur audio avec un codeur audio par transformée de l'état de l'art, le codeur AAC de iTunes 7. Nous comparons aussi avec une version de notre codeur sans modèle psychoacoustique. Les résultats montrent que le modèle psychoacoustique donne un gain significatif et permet des performances comparables avec celles du codeur AAC.

Evaluation subjective Nous proposons dans un second temps une évaluation subjective du codeur audio proposé. Nous avons réalisé un test MUSHRA qui a impliqué une vingtaine de participants. Ce test compare les performances de 3 codeurs à deux débits, et pour 5 signaux différents. Les résultats montrent premièrement que le modèle psychoacoustique permet un gain conséquent mais uniquement pour les signaux polyphoniques. Les résultats montrent ensuite que les performances de notre codeur sont bien meilleurs que le codeur AAC pour les sons monophoniques, mais légèrement inférieurs pour les sons monophoniques.

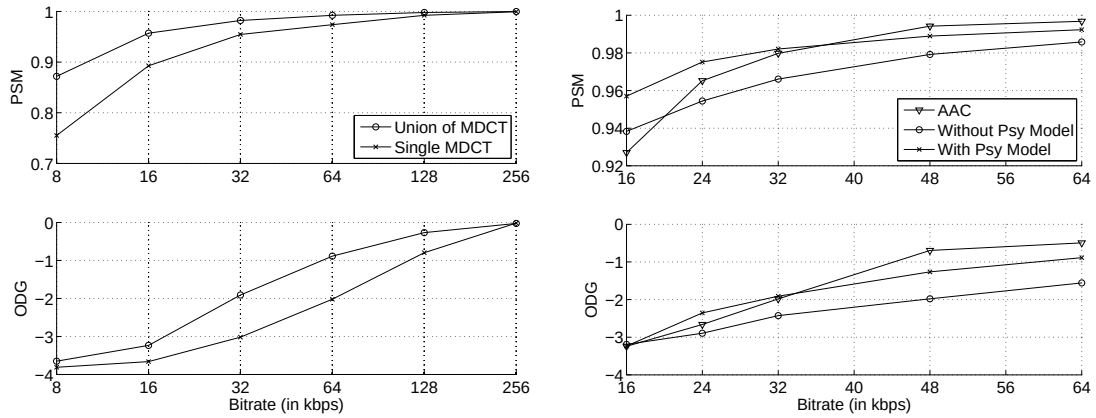

Figure 0.5: *Résultats de l'evaluation objective avec PEMO-Q.*

Indexation audio dans le domaine transformée

Nous proposons dans la suite une étude de l'indexation audio dans le domain transformée. Nous proposons des algorithmes simples pour le calcul de représentations mi-niveaux directement à partir de la représentation interne d'un codeur audio. Nous considérons trois applications: le suivi de rhythme, la reconaissance d'accords, le classification de genre musicale. Nous comparons les résultats obtenus avec trois codeurs: deux codeurs de l'état de l'art (MP3 et AAC), et un codeur proposé dans cette thèse.

Codage audio et représentation du signal

Nous étudions dans la suite trois codeurs audio: le MPEG-1 Layer 3 (MP3), le MPEG-4 Advanced Audio Coding (AAC), et un codec semblable à celui proposé précédemment mais avec le Matching Pursuit standard (sans contrôle de pré-echo) et sans modèle psycho-acoustique (ce codec est noté 8xMDCT).

MP3 Le MP3 utilise un banc de filtres hybride: un banc de filtre Polyphase Quadrature Filterbank (PQF) à 32 sous-bandes+une MDCT adaptative dans chaque sous-bande. Certains travaux existants utilisent les signaux en sous-bandes du banc de filtres PQF, nous préférons utiliser les coefficients MDCT pour deux raisons: une meilleure résolution fréquentielle, un coût calculatoire réduit (pas de transformée inverse).

AAC Nous utilisons ici la version la plus simple du codeur AAC, il s'agit du profil Low Complexity (LC) sans Temporal Noise Shaping (TNS) ni Perceptual Noise Substitution (PNS). La représentation du signal est alors très simple car composée d'une simple MDCT adaptative. Nous utiliserons dans la suite les coefficients de cette MDCT.

8xMDCT Le codeur proposé utilise une union de 8 bases MDCT. Cette représentation du signal a déjà été décrite précédemment. Nous utiliserons dans la suite les coefficients de ces 8 MDCT.

Représentations mi-niveaux

Nous proposons ici des algorithmes pour le calcul de 3 représentations mi-niveaux dans le domaine transformée: une fonction de détection d'attaque, un chromagramme, des coefficients

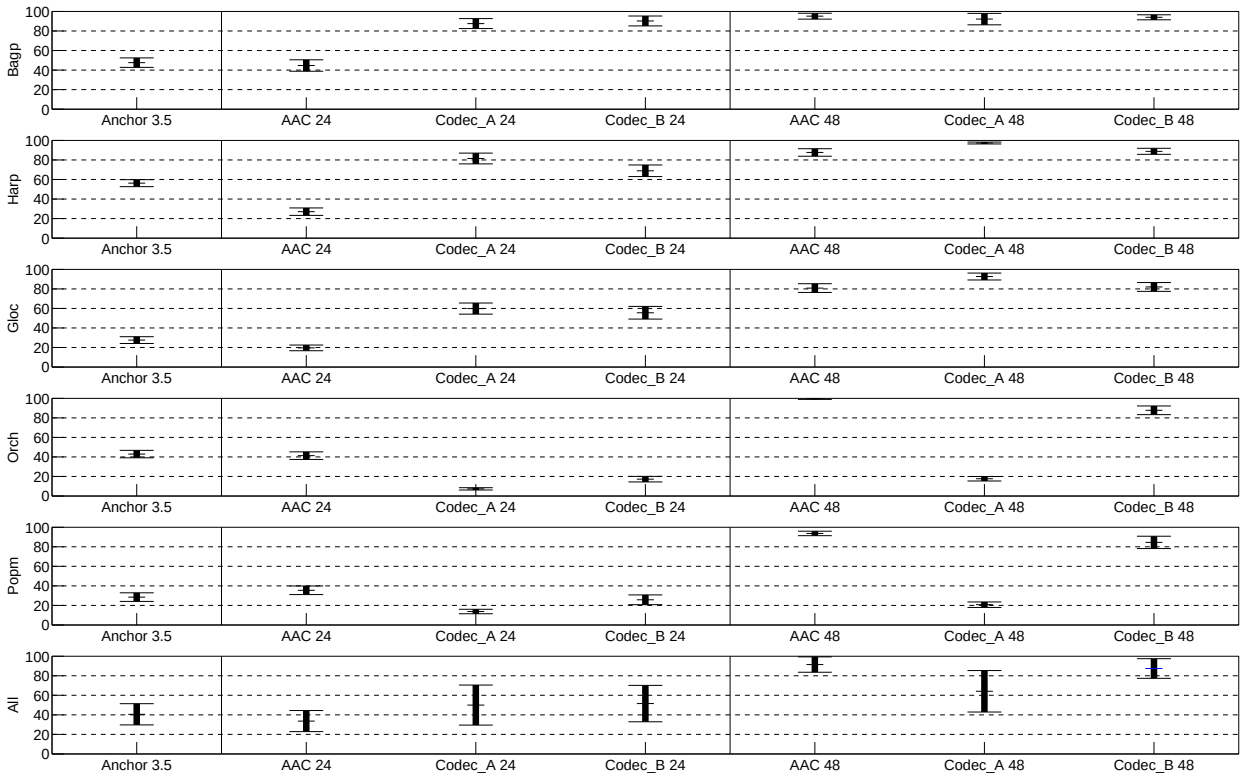

Figure 0.6: *Résultats du test MUSHRA.*

MFCC.

Fonction de détection d'attaque Pour les codeurs MP3 et AAC, la fonction de détection d'attaque est calculée de manière similaire au calcul du flux spectral où l'amplitude de la transformée de Fourier à court-terme est remplacé par l'amplitude des coefficients MDCT. Pour le codeur 8xMDCT, l'approche est différente, on ne garde que les coefficients des 2 MDCT à plus petite taille de fenêtre et on somme l'amplitude de ces coefficients dans des "cases" temporelles.

Chromagramme La faible résolution fréquentielle des codeurs MP3 et AAC empêche le calcul d'un chromagramme. Par contre, l'utilisation de MDCT avec des grandes fenêtres dans le codeur 8xMDCT permet le calcul d'un chromagramme performant. Pour cela, on ne garde que les coefficients des 2 MDCT à plus grande taille de fenêtre et on somme l'amplitude de ces coefficients dans des "cases" temps/fréquence.

Coefficients MFCC Pour les codeurs MP3 et AAC, les coefficients MFCC sont calculés de manière similaire au calcul dans le domaine temporelle où l'amplitude de la transformée de Fourier à court-terme est remplacé par l'amplitude des coefficients MDCT. Pour le codeur 8xMDCT, on utilise une approche multi-échelle similaire à l'approche initialement proposé par Marcela Morvidone, on calcule un histogramme pondéré fréquence/échelle puis on applique une DCT à deux dimensions.

Applications

Nous proposons l'évaluation des méthodes proposées pour 3 applications: suivi de rythme, reconnaissance d'accords, et classification de genre. Le suivi de rythme utilise la fonction de

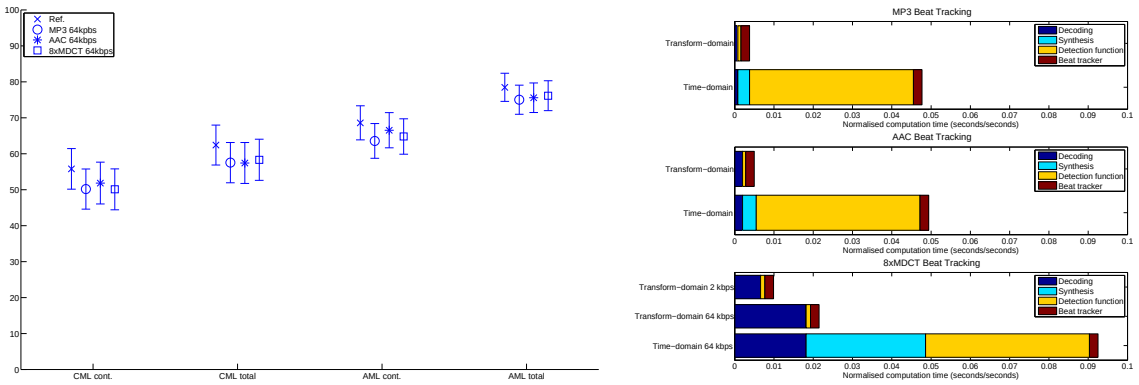

Figure 0.7: *Suivi de rythme: performance et temps de calcul.*

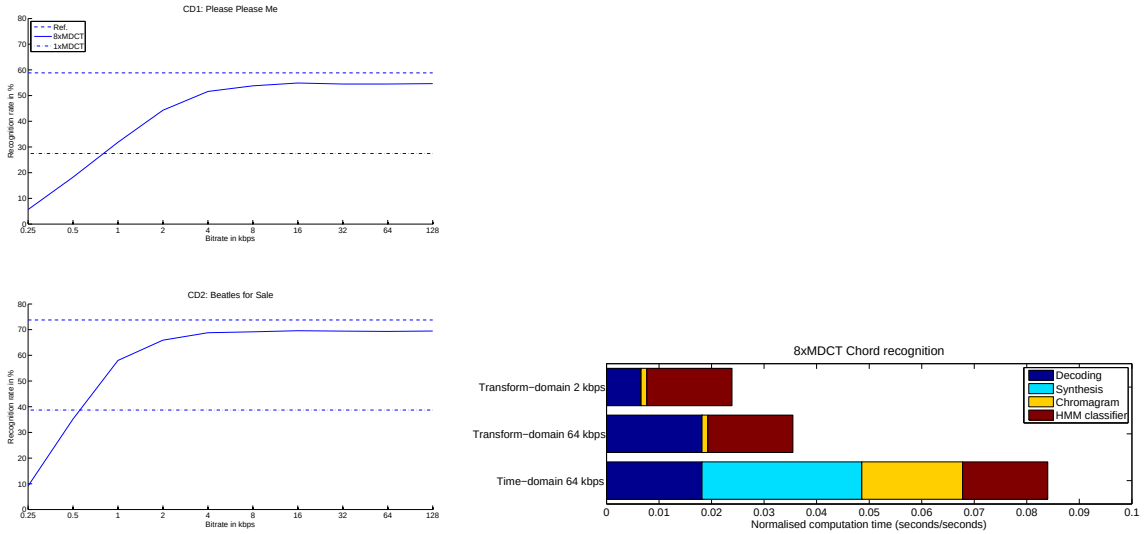

Figure 0.8: *Reconnaissance d'accords: performance et temps de calcul.*

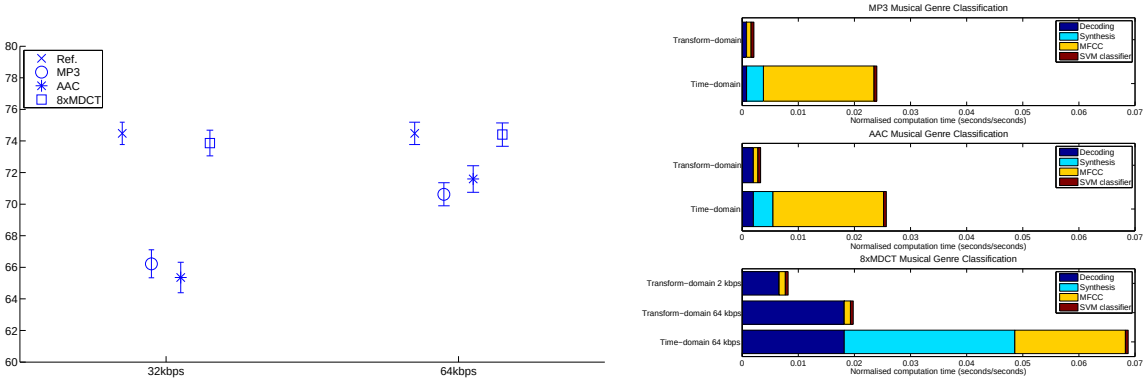

Figure 0.9: *Classification de genre musical: performance et temps de calcul.*

détection d'attaque et un système proposé par M. Davies. La reconnaissance d'accords utilise le chromagramme et un système proposé par J. Bello. La classification de genre utilise les coefficients MFCC et un système basé sur des machines à vecteur support (SVM). Les figures ci-dessous donnent quelques résultats pour les systèmes dans le domaine transformée ainsi que pour les systèmes de référence dans le domaine temporelle.

Conclusion

Conclusion générale

La motivation initiale de cette thèse était l'étude de l'application de nouvelles techniques de représentation du signal au codage audio. Ces nouvelles techniques sont basées sur un domaine de recherche appelé "représentations parcimonieuses de signaux". Elles permettent la conception de représentations du signal qui ont plusieurs avantages comparés aux approches traditionnels, comme une meilleure résolution ainsi qu'une meilleure flexibilité.

Cependant, le principale inconvénient de ces méthodes est leur complexité calculatoire. Quand ces techniques ont été développés (ex. Matching Pursuit en 1992), ce problème de complexité empêchait leurs utilisations dans des applications concrètes. Ce n'est que depuis quelques années, avec l'augmentation des performances des ordinateurs et aussi surtout le développement d'algorithmes performants (ex. MPTK), que ces techniques ont trouvé un intérêt dans certaines applications. Par exemple, un des systèmes de codage que nous proposons obtient des temps calcul d'environ 3 fois le temps réel, ce qui est acceptable pour certaines applications.

Bien sûr, les méthodes proposées sont toujours plus lentes que les approches traditionnels utilisés en codage par transformée, mais nous avons montrés que notre approche a plusieurs avantages par rapport au codage par transformée, et par conséquent le compromis performance/complexité à maintenant tendance à pencher en notre faveur. Ces avantages comprennent de meilleures performances à bas débit, et une qualité transparente à haut débit. Notre approche permet donc un "codeur plus universelle" que le traditionnel codage par transformée.

Cependant, les méthodes proposées ne sont encore qu'une première étape vers un codeur vraiment universel et performant, car nous avons vu que les performances obtenues par notre système sont élevées uniquement pour les sons monophoniques. Pour les sons polyphoniques, les performances ne sont pas aussi satisfaisantes, mais nous avons aussi montrés que, dans ce cas-là, notre approche a un autre avantage par rapport au codage par transformée qui est une indexation audio dans le domaine transformée performant. Nous avons montré que combiner codage audio et indexation audio est maintenant possible.

Perspectives de recherche

Nous proposons ci-dessous plusieurs perspectives de recherche.

Représentation du signal Une première extension serait d'approfondir l'approche basée sur la MCLT introduite dans le chapitre 7. Une deuxième extension serait d'étudier d'autres modèles de signaux comme l'approche moléculaire introduite dans la thèse de Pierre Leveau. Une troisième extension serait d'étudier d'autres algorithmes de décomposition, comme le Gradient Pursuit. Une étude préliminaire montre que le Gradient Pursuit obtient de meilleures performances que le Matching Pursuit avec un coût calculatoire comparable.

Codage audio Une première extension serait d'étudier d'autres algorithmes de codage de source pour le codage en plan de bits, comme des algorithmes de codage arithmétique à contexte ou des algorithmes basés modèles comme ceux étudiés dans la thèse de Marie Oger. Une deuxième

extension serait d'étudier de meilleures modèles pyschoacoustiques pour les représentations multi-échelles. D'autres possibilités sont l'étude de la MDCT entière pour le codage sans pertes, et l'extension au stéréo et à la réplication de bande.

Indexation audio Une première extension serait d'étudier d'autres codeurs standards comme le MPEG-4 HILN ou le MPEG-4 SSC. Une deuxième extension serait d'étudier le codec proposé dans le chapitre 6. Une dernière extension serait d'étudier d'autres applications, comme la recherche du plus proche voisin dans une base de fichier codés, une application déjà étudié dans le cadre du codage d'image.

Acknowledgments

First of all, I would like to thank all my thesis jury: Laurent Daudet, Mike Davies, Pierre Duhamel, Jean-Gabriel Ganascia, Bastiaan Kleijn, Pierrick Philippe and Gaël Richard. I thank you for carefully evaluating my thesis and for your pertinent comments and interesting questions.

I would like particularly to thank my supervisors Laurent Daudet and Gaël Richard without whom all this work could not be possible. I thank them for their support, their help, their ideas, their guidance, the motivation and the freedom they gave me during these three years.

I would like to thank all the researchers I had the pleasure to work with: Rémi Gribonval and Sacha Krstulovic from IRISA, for their help in understanding and contributing to the MPTK project, I thank them particularly for the several days I spent in Rennes; Pierre Leveau, a past PhD Student in LAM, with whom I had interesting discussions and collaborations; the intern Grégory cornuz who worked with Pierre Leveau on object-based audio coding; Pierre Vandergheynst and his team, who kindly invite me during two months in Lausanne, I thank you for the time you spent with me; Olivier derrien, with whom I had useful discussions on transform audio coding and psychoacoustic models; Marcela Morvidone, a past postdoctoral scholar in LAM, who greatly contributes to the multi-scale MFCC-like I use in this thesis; Matthew Davies from QMUL and Juan Bello from NYU, I thank you for your source code and your help on beat tracking and chord recognition; Nicolas Montgermont, my office-mate; all members of the LAM team, and particularly its director Jean-Dominique Polack.

I would like to take advantage of the opportunity to thank all my past supervisors, who greatly contributed to the passion I have in research: Jean-Yves Tourneret from TéSA (Toulouse), who gave me the taste of signal processing research; Philippe Gournay and Roch Lefevbre from Sherbrooke University, who make me discover the world of audio coding; Juan Bello and Mark Sandler from QMUL, who confirmed my passion for digital music.

I also would like to thank all the institutions that supported me materially during these 3 years. The french ministry for higher education and research and the university Pierre and Marie Curie for the grant that funded my PhD studies. I particularly thank the personns who help me getting the grant: Laurent Daudet who help me writing the request; Roch Lefevbre, Mark Sandler and Francis Castanié for their support letters. The institutions that funded several travels and conferences: QMUL, IRISA, Sherbrooke University, Edinburgh University, TELECOM ParisTech, EPFL, LaBRI, CNRS...

Finally, I would like to thank my family and my friends for their support. I particularly thank my wife Linh, who was always there for me.

Contents

Chapter 1

Introduction

1.1 Background

The last decades have seen a revolution in the way sound is processed. Sound, which can be defined as a pressure wave transmitted through a medium (e.g. the air), was originally processed as a continuous (or analog) signal. A typical analog audio system could be described as follows. A transducer (e.g. a microphone) is used to convert the pressure wave into a continuous signal (e.g. an electrical signal), which is then stored (e.g. on a compact cassette) or transmitted (e.g. on a FM channel). At the end of the chain, the continuous signal is restored, amplified and converted into a pressure wave using e.g. a loudspeaker. The first devices able to record and to play back sound appeared at the end of the 19th century (invention of the phonograph by Thomas Edison in 1877) but the analog audio devices became widely used only later, with the spread of the turntable in the 1920s and the spread of the compact cassette in the 1960s.

With the increase of the computing hardware performance in the 70s and later, analog audio systems have gradually been replaced by digital audio systems. A typical digital audio system could be described as follows (see Fig. 1.1). As in an analog audio system, a transducer is first used to convert the pressure wave into an electrical signal. Then, this analog signal is converted into a digital signal using a three-step process, called Pulse Code Modulation (PCM): the continuous signal is first sampled at a given sampling rate, then each sample is quantized at a given resolution, and finally converted into a binary format composed by a sequence of 0 and 1, called bits. This digital signal is then stored (e.g. on a hard disk), or transmitted (e.g. through the internet). At the end of the chain, the digital signal is restored, converted into an electrical signal, and played back as an analog signal. Digital audio has several advantages over analog audio including high robustness to channel noise and interferences, lossless duplication, powerful processing capabilities, and cheap high-quality hardware. Digital audio became widely used with the spread of the Compact Disc (CD) in the 80s, replacing the traditional vinyl record and the compact cassette. The CD is still

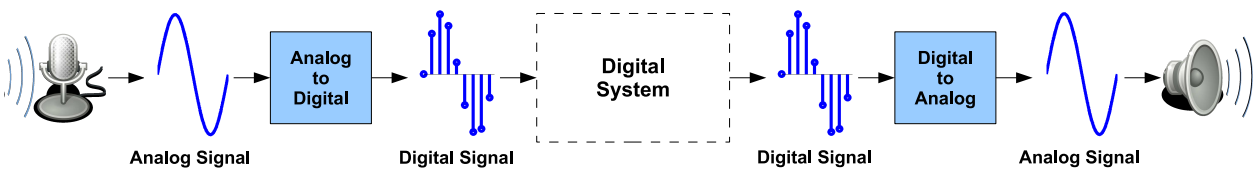

Figure 1.1: *A typical digital audio chain.*

considered as the highest quality format widely available on the market[1]: its format is based on a 2-channel (stereo) track with a sampling rate of 44100 samples per second where each sample is quantized on 65536 values (i.e. 16 bits per sample).

Digital audio has many advantages over analog audio, and these made him preferable in most applications. However, the traditional way of representing a digital signal using PCM has a main drawback which is a high storage requirement. As an example, a 12-minute stereo track in CD format requires around one billion bits. To lower costs related to the storage and transmission of digital audio data, there was then a necessity to develop methods that allow the representation of digital audio with fewer bits than PCM. This research area, known as audio coding, emerged in the 80s with the development of technologies based on digital signal processing techniques and psychoacoustic models. Fig. 1.2 shows a simplified block diagram of an audio codec. An audio codec is the combination of an audio coder and an audio decoder. The audio coder takes as input a (digital) audio signal and outputs a sequence of bits using a two-stage approach: the input audio signal is first transformed into a different domain (e.g. time-frequency), then the resulting coefficients or parameters that describe this signal representation are converted into a sequence of bits using a source coding technique. The audio decoder has the same two-stages configuration as the coder: it first converts the sequence of bits into a set of coefficients or parameters, and then synthesizes the decoded audio signal using these coefficients or parameters.

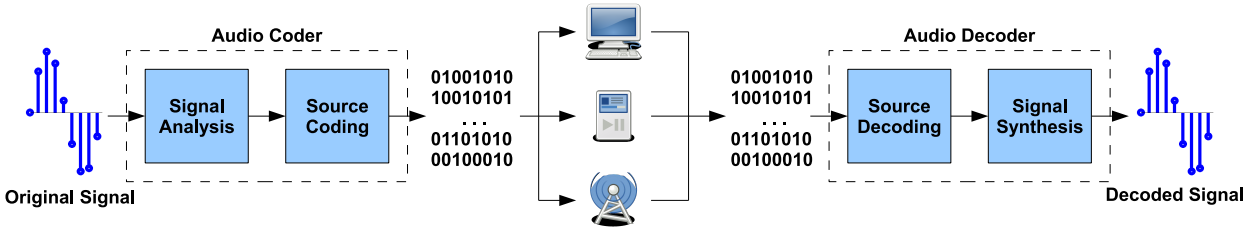

Figure 1.2: *A simplified audio codec (assuming lossless transmission/storage)*

The basic principle of audio coding is to reduce statistical redundancy and perceptual irrelevancy in an audio signal. On the one hand, statistical redundancy is reduced by exploiting the properties of the source of the sound, which is actually the ultimate first stage of the digital audio chain. There are mainly two types of statistical redundancy. Firstly, the consecutive samples of an audio signal are generally highly dependent; this property is exploited by the first stage of the audio coder (i.e. the signal representation technique) using e.g. a time-frequency transform. Secondly, the coefficients or parameters that describe the signal representation have generally a highly non-uniform statistical distribution; this property is exploited by the second stage of the audio coder (i.e. the source coding technique) using e.g. Huffman coding. On the other hand, perceptual irrelevancy is reduced by exploiting the properties of the human auditory system, which is actually the ultimate last stage of the digital audio chain. Two main properties of the auditory system are used in audio coding, the absolute threshold of hearing and auditory masking. The absolute threshold of hearing is defined as the sound level below which a pure tone is not perceived by the auditory system, while auditory masking is defined as the increase of the threshold of audibility of one sound in the presence of another sound. These two properties are exploited in audio coding by shaping the coding noise such that it is below the threshold of audibility. This is generaly done using a psychoacoustic model computed on the input audio signal and that controls the source

[1]It is worth noting that the market proposes also the Super Audio CD (SACD), which is of higher quality but it is not widely used.

coding technique (the second stage of the coder). The coefficients or parameters that describe the signal representation are then coded with a precision that depends of the masking threshold computed by the psychoacoustic model. In some cases, perceptual irrelevancy is also reduced in the first stage of the audio coder using a signal representation technique controled by a psychoacoustic model (e.g. a sinusoidal model that only extracts the most perceptually important sinusoids).

In early 90s, prestigious labs including AT&T Bell Labs, Fraunhofer IIS, Thomson-Brandt and France-Telecom-CCETT gathered their technologies and gave birth to the Moving Picture Expert Group 1 (MPEG-1) standard, published in 1993. This standard is based on 3 layers of increasing quality and complexity, including the well known MPEG-1 Layer 3 (MP3). These are all based on a subband approach where the input digital signal is first mapped into a time-frequency representation composed by several subband signals, which are then encoded according to a psychoacoustic model. The MP3 format at a bitrate of 128 kilobits per seconds (one minute of stereo music approximately equal to eight million bits) is able to represent sound with almost the same quality as a CD but with 12 times less bits. This property made MP3 very attractive when it appeared on the market because the size of the personal computer hard disks rarely exceeded several hundred Megabytes at that time. Moreover, MP3 format really began to spread widely with the development of internet in the second half of the 90s. Napster, the first peer-to-peer network released in the 1999, is a well-known example of the success of the MP3 format. The number of users that shared MP3 files on the Napster network was estimated at approximately 25 million in 2001, just before its shutdown due to a lawsuit issued by the Recording Industry Association of America.

Despite of the success of the MP3 format, which is still widely used all over the world, research in audio coding still continued in the 90s. Several improvements over the MP3 gave birth to a new standard called Advanced Audio Coding (AAC), first introduced in the MPEG-2 standard in 1997 and included in the MPEG-4 standard in 1999. Though the basic principle of AAC is based on a similar subband approach as MP3, this second-generation audio codec allows more flexibility and better performance than MP3. It is generally considered that AAC at 96 kbps gives approximately the same near-CD quality as MP3 at 128 kbps. However, AAC did not get the same success as MP3 at first, it started to become widespread only in 2003, when Apple announced that its music online store (iTunes store) and its mobile device products (iPod) would support the AAC format. At that time, music songs was sold in a proprietary Digital Rights Management (DRM)-restricted form of AAC, to avoid the copyright-related problems raised by Napster. Despite this limitation, iTunes store proved rapidly its success and viability with 200 million songs sold at the end of 2004, 1.5 billion songs at the end of 2006, and 5 billion songs at the mid of 2008.

AAC is still considered as the state-of-the-art audio coding format for CD-quality music, and there is probably very little room for improvement now in this context. However, it is known that encoding stereo music with AAC at low bitrates (typically lower than 96 kbps) gives poor quality, far from the reference CD-quality. This issue has motivated several research teams to propose new low-bitrate audio coding technologies that perform better than AAC at low bitrates. These third-generation audio codecs include High Efficiency-AAC (HE-AAC) standardized in MPEG-4 in 2003 and revised (HE-AACv2) in 2005, SinuSoidal Coding (SSC) standardized in MPEG-4 in 2004, and Extended Adaptive Multi-Rate Wideband (AMR-WB+) standardized by the 3rd Generation Partnership Project (3GPP) in 2004. Contrary to the MP3 and AAC codecs which are based on subband techniques, these new audio codecs use parametric modeling. This includes spectral band replication (HE-AAC, HE-AACv2 and AMR-WB+), sinusoidal+transients+noise modeling (SSC), linear prediction (AMR-WB+) and parametric stereo (HE-AACv2, SSC and AMR-WB+). Listening tests showed that HE-AACv2, SSC and AMR-WB+ allow much better quality than AAC

at bitrates of 48 kbps and even lower. It is important to note that these new audio codecs are not a replacement for AAC but rather a complement because they are targeted for low bitrate coding only and can't give the same CD-quality as AAC at high bitrates. They find thus interests in applications that require low bitrates such as 3G mobile devices, streaming applications, and digital broadcasting. Though these audio coding formats are not yet as spread as MP3 and AAC, they are more and more widely used.

There are now many efficient and standardized audio codecs on the market, but each one has been designed for a specific target. Indeed, the performance of each codec strongly depends on several parameters including the input audio material (e.g. speech, monophonic music, polyphonic music...), the application constraints (e.g. bitrate, quality, delay, complexity...) and possibly the network constraints (e.g. internet, switched telephone network...). A current research challenge in audio coding is then to design universal audio coding technologies that combine the advantages of existing approaches in a flexible and adaptive way. Universal audio coding would have many interests including simplified systems (a single codec instead of one codec for each application constraint), adaptation to user preferences (e.g. reduced complexity for reduced power consumption), efficient use of available bandwidth or storage capacity (codec adaptation to application/network constraints in real time) and possibly transcoding (e.g. transcoding between different bitrates). There is currently an active research community that investigates this research area, e.g. the european projects ARDOR (adatpive rate-distortion optimised sound coding) and FLEXCODE (flexible coding for heterogeneous networks).

Another challenge in audio coding, much less explored, is to combine audio coding with audio indexing (i.e. audio content-based automatic indexing). Audio indexing is useful for music information retrieval (MIR), a recent research domain that studies the problem of finding quickly a given information in the constant increasing mass of digital music data. MIR includes several useful tasks such as e.g. chord transcription, rhythm analysis, musical genre classification, cover-song identification, music recommendation, playlist generation, audio fingerprinting... Though researches in audio coding and audio indexing have been conducted independently, the methods used in both areas have many similarities. Then, the challenge would be to design a single technique that is useful for both audio coding and audio indexing. Such a technique would be used by an audio coder at the analysis stage. This would have for consequence that the sequence of bits produced by the coder contains most of the information necessary to perform audio indexing tasks. The final audio indexing application is then performed at the decoder with very low additional computation (see Fig. 1.3). A system that combines audio coding and audio indexing would have obvious interests for applications that require low computational costs, such as processing on mobile devices, or processing very large databases of coded files.

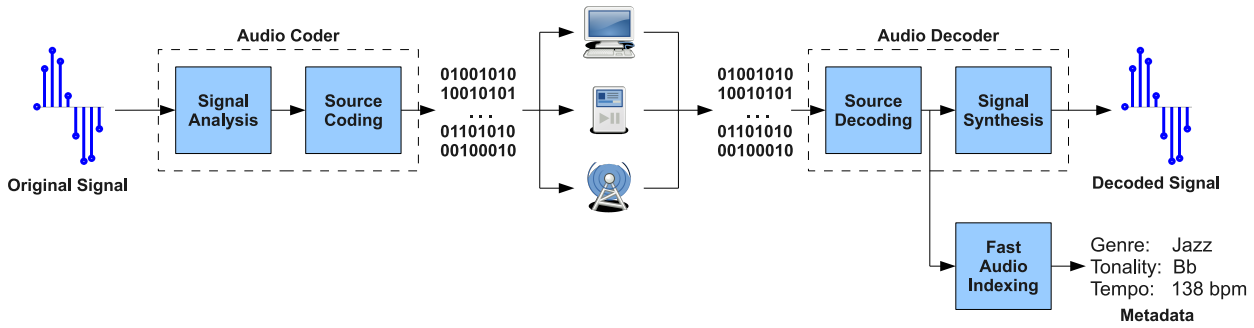

Figure 1.3: *An audio codec with a transform-domain audio indexing system*

1.2 Main contributions

The main contributions of the thesis can be summarized as follows

New signal representation approaches We propose in this thesis new signal representation approaches that are useful for both audio coding and transform-domain audio indexing. These novel methods are based on a research domain known as sparse signal representations and that aim at modeling a signal as a sum of small number of elementary functions which are chosen among a collection of arbitrary size (usually much larger than the signal dimension). Contrary to the traditional approach in high-quality audio coding where a signal is represented using a time-frequency transform (e.g. AAC), the proposed method uses an overcomplete union of several transforms with different time-frequency tradeoffs. This allows the modeling of a signal with a fewer number of elementary functions than in the transform case, a property obviously useful for audio coding. The main issue is to find efficient ways to decompose a signal in this overcomplete set of functions. We thus propose several algorithms based on the matching pursuit algorithm, namely a fast implementation, a pre-echo control modification, and a rate-distortion optimization. All these algorithms have been implemented in C++ and are freely available[2] (GNU General Public License). It is worth noting that the proposed study also investigates the problem of choosing the set of elementary functions (also called dictionary), which is a fundamental problem for sparse signal representations.

Improvements in audio coding We propose in this thesis two audio codecs based on the new signal representation approaches. The main issue is to find efficient and flexible coding strategies for the proposed signal representations. We thus propose several new techniques that allow efficient and high-quality scalable audio coding, namely a coefficient interleaving process, a psychoacoustic bitplane coding algorithm and an adaptive bitplane coding algorithm. The first audio codec uses the psychoacoustic bitplane coding algorithm; it is evaluated using PEMO-Q, a recent objective measure developed at Oldenburg university, and a MUSHRA listening test performed in our lab. This codec has been implemented in C++ and is freely available[2] (GNU General Public License). The second audio codec uses the adaptive approach; we provide only preliminary results in this case as we have not yet performed a complete evaluation. We also propose in this thesis a study on embedded polar quantization that would be useful for e.g. scalable audio coding based on complex transforms (e.g. the modulated complex lapped transform), this study is however still preliminary as we do not provide a complete audio codec based on this approach.

Improvements in transform-domain audio indexing We propose in this thesis new approaches for transform-domain audio indexing. We consider both new audio codecs and new applications. Existing prior work is relatively limited, only a few results have been published in the literature for the MPEG-1 Layer 3 format only. In our work, we study not only the MP3 format but also AAC and one of the codec we have developed in the thesis. For each codec, we propose fast algorithms that compute features for audio indexing tasks. Three applications are considered: beat tracking, chord recognition and musical genre classification. For each codec and for each application, we provide evaluation of both the performance and the computation time. The proposed algorithms have been implemented in C++ (using existing open source softwares for MP3/AAC decoding) and are freely available[2] (GNU General Public License).

[2]Download source code at the following address: `http://www.emmanuel-ravelli.com/downloads`

1.3 Thesis outline

The main contributions are organized in the thesis as shown in Fig. 1.4. It is important to note that the chapters are sufficiently self-contained that they can be read independently. We detail the content of each chapter below.

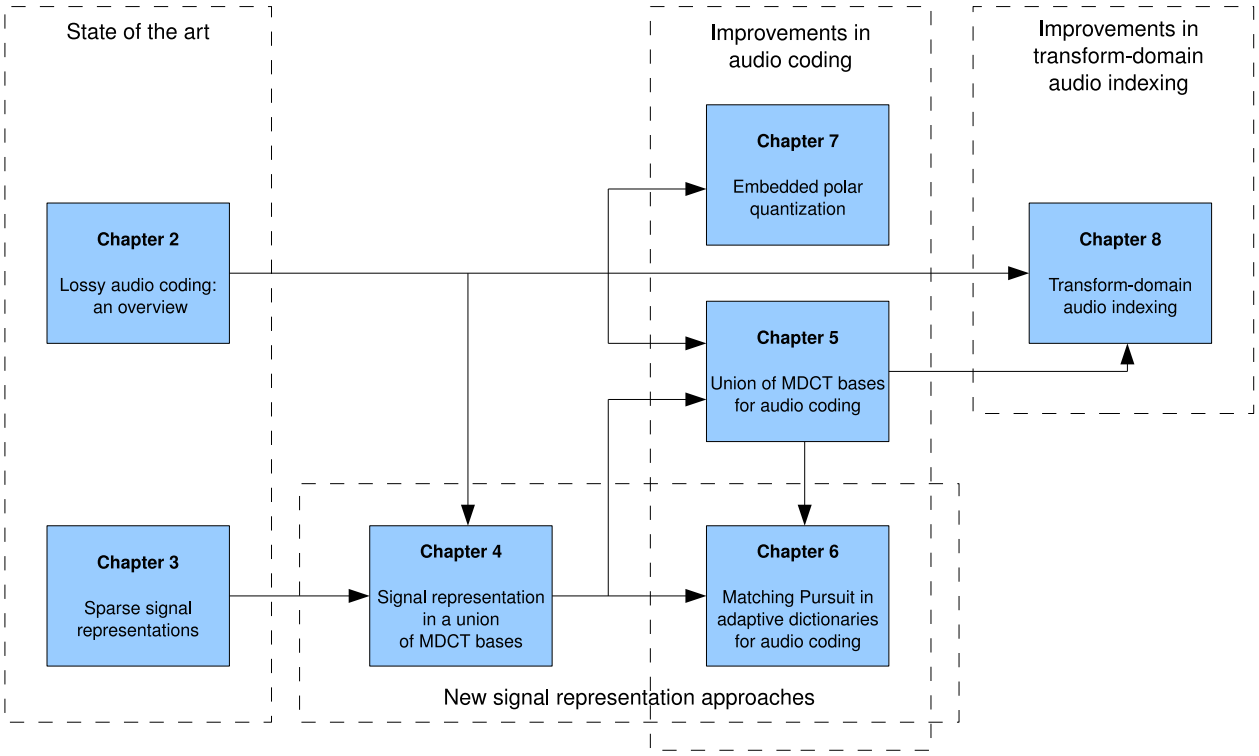

Figure 1.4: *Thesis organization*

Chapter 2 proposes an overview of lossy audio coding. This chapter first reviews existing signal representation methods used in modern lossy audio coding, then describes briefly the main approaches used for source coding and finally presents a few methods for both subjectively and objectively evaluating lossy audio codecs.

Chapter 3 proposes a brief review of sparse signal representations, a class of signal representation methods used in several coding applications (including audio, image and video). This chapter first introduces the mathematical problems and some algorithms to solve them, then describes several signal models used in coding applications.

Chapter 4 introduces a new signal representation approach based on sparse signal representations. This chapter first describes the motivations and the signal model, then proposes several algorithms to approximate a signal with this model, and finally presents the results.

Chapter 5 proposes a source coding algorithm that efficiently encodes the new signal representation approach presented in the Chapter 4. This chapter gives results for each stage of the proposed codec, and proposes an objective and subjective evaluation of the final codec. Results presented in chapters 4 and 5 have been published in two papers: one presented at WASPAA 2007 [RRD07] and one accepted for future publication in IEEE Transactions on Speech, Audio and Language Processing [RRD08c].

Chapter 6 introduces a way to improve the audio codec presented in the previous chapter.

This chapter proposes improvements for both the signal representation and the source coding. This chapter also gives preliminary results of the improved audio codec, both in terms of performance and complexity. Results presented in this chapter have been published in a paper that will be presented at EUSIPCO 2008 [RRD08b].

Chapter 7 proposes a short study on embedded polar quantization. This chapter proposes simple algorithms for the design of embedded quantizers for circularly symmetric data. This could be applied for example to sparse signal representations based on complex transforms. One notes that we do not propose an audio codec in this chapter, we only study the quantization stage. This chapter has been published in IEEE Signal Processing Letters [RD07].

Chapter 8 studies transform-domain audio indexing. This chapter proposes simple algorithms for the computation of several mid-level representations for transform-domain audio indexing. This chapter considers three applications, respectively beat tracking, chord recognition and musical genre classification. The performance of three coders are compared and discussed. Two standard filter-bank based audio codecs and one of our audio codec are considered. A subset of the results presented in this chapter have been published in a paper that will be presented at ISMIR 2008 [RRD08a], and a longer journal paper is currently being written.

Chapter 9 proposes conclusions and possible future work.

Chapter 2

Lossy audio coding: an overview

Abstract

We propose in the first chapter a state-of-the-art of lossy audio coding, which is the main topic of this thesis. The aim of this chapter is not to give an exhaustive review of existing technologies as the literature on audio coding is very wide. We rather review the main techniques used in the state-of-the-art audio codecs such as those from MPEG and 3GPP. We also emphasize several techniques that we will use in the rest of this thesis such as the Modified Discrete Cosine Transform and the bitplane coding.

Contents

2.1 Introduction

The basic principle of lossy audio coding is to reduce statistical redundancy and perceptual irrelevancy in an audio signal. To perform such a task, an audio coder uses two main operations (see Fig. 2.1). First, the original audio signal is analyzed using a signal representation technique. Then, the resulting coefficients or parameters that describe this signal representation are converted into a binary format using a source coding technique. Depending on the techniques used, each stage of the audio coder removes either statistical redundancy and/or perceptual irrelevancy. The audio decoder has the same two-stages configuration as the coder, it first converts the sequence of bits into a set of coefficients or parameters, and then synthesizes the decoded audio signal using these coefficients or parameters.

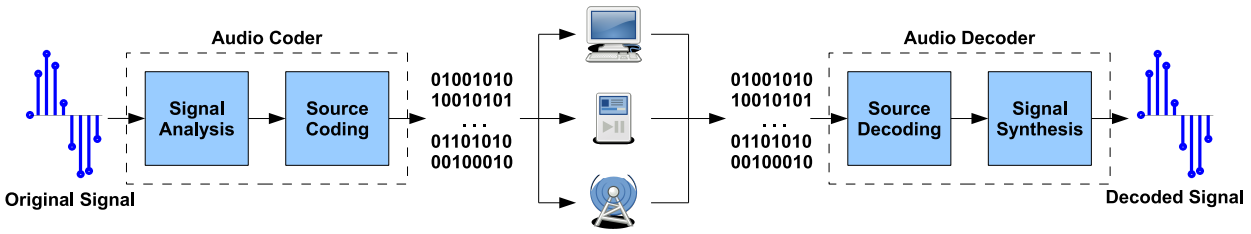

Figure 2.1: *A simplified audio codec (assuming lossless transmission/storage)*

In this chapter, we review some of the most widely used techniques in lossy audio coding: some signal representation techniques in section 2.2, and some source coding techniques in section 2.3. Finally, as it is necessary to evaluate the degradations introduced by a codec in lossy audio coding, we review some evaluation techniques in the section 2.4.

2.2 Signal representation

There are three main categories of signal representation used in modern lossy audio coding: the filter bank approaches, the parametric models and the hybrid approaches. The basic principle of the filter bank approaches is to map a temporal domain signal into a time-frequency representation composed by several subband signals. These are used in CD-quality audio codecs such as MP3 and AAC. Contrary to the filter banks, the parametric modeling approaches represent, generally, only a certain subspace of the signal using a few parameters. These are used in low-bitrate speech and audio codecs such as AMR and SSC. Finally, the hybrid approaches combine filterbank-based and parametric-based techniques for improved performance. They are used in recent audio coders such as HE-AAC and AMR-WB+. We detail in this section the most popular techniques in these three categories.

2.2.1 Filter bank approaches

The filter bank approach is historically one of the first signal representation technique used in audio coding and it is now one of the most widely used. There are two main reasons that make the filter banks attractive for audio coding. First, the filter bank reduces the redundancy of an audio signal, by mapping the time domain signal into subband signals where only a few of these subband signals have a high energy. Second, this time-frequency mapping allows an individual

processing of each subband signal, which is adapted to the perceptually-based quantization used in audio codecs.

A K-band system is given in Fig. 2.2. It is composed of two filter banks, an analysis and a synthesis filter bank. In the coder, the analysis filter bank maps the time-domain signal into K subband signals. These subband signals are then encoded. In the decoder, the subband signals are decoded and passed through the synthesis filter bank in order to restore the time-domain signal. A common K-band analysis-synthesis system used in audio coding has the following two properties. First, the analysis-synthesis system is critically sampled: the overall samplerate of the subband signals is the same than the samplerate of the original signal (We will see later in this thesis that it is possible in audio coding to use filter banks which are not critically sampled). This is easily done using downsampling and upsampling as shown in Fig. 2.2. Second, the analysis-synthesis system allows perfect or near-perfect reconstruction (up to a delay), a property more difficult to achieve.

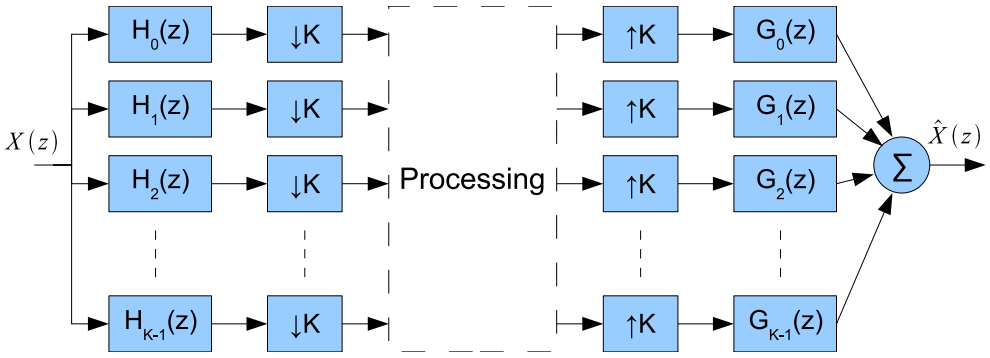

Figure 2.2: *A K-band analysis/synthesis filter bank used in audio coding.*

The common approach to achieve a K-band critically sampled filter bank with (near-)perfect reconstruction is to use a low pass FIR prototype filter $H(z)$ and define the analysis filters $H_k(z)$ and the synthesis filters $G_k(z)$ as a function of this prototype filter. Thus, the design of the filter bank depends only on the design of a FIR prototype filter. We will assume in the following that all filters have the same length, written L (in samples).

We describe in the following the two most widely used filter bank approaches in audio coding, the K-band Polyphase Quadrature Filter bank (PQF) and the Modified Discrete Cosine Transform (MDCT).

K-band Polyphase Quadrature Filter bank

The K-band Polyphase Quadrature Filter bank (PQF) has been proposed independently by Nussbaumer in 1981 [Nus81] and Rothweiler in 1983 [Rot83]. It is also referred sometimes as Pseudo Quadrature Mirror Filter bank (PQMF) in the literature. The filters of the PQF are defined in the temporal domain as:

$$h_k(n) = h(n)cos\left[\frac{\pi}{K}\left(k+\frac{1}{2}\right)\left(n-\frac{L-1}{2}\right)+\theta_k\right] \quad (2.1)$$

$$g_k(n) = h(n)cos\left[\frac{\pi}{K}\left(k+\frac{1}{2}\right)\left(n-\frac{L-1}{2}\right)-\theta_k\right] \quad (2.2)$$

$$(2.3)$$

where $h(n)$ the impulse response of the low-pass FIR prototype filter and L the length of the filters. To achieve perfect reconstruction, the θ_k must verify the following constraints [Rot83]

$$e^{j4\theta_0} = -1 \tag{2.4}$$

$$e^{j2\theta_k} + e^{j2\theta_{k-1}} = 0 \text{ , with } 1 \leq k < K \tag{2.5}$$

$$e^{j4\theta_{K-1}} = -1 \tag{2.6}$$

and the transfer function $H(z)$ of the prototype filter must verify the following constraints:

$$H\left(e^{j\omega}\right) = 0 \quad \text{if} \quad \pi/K \leq |\omega| \tag{2.7}$$

$$\left|H\left(e^{j\omega}\right)\right|^2 + \left|H\left(e^{j(\pi/K-\omega)}\right)\right|^2 = cst \quad \text{if} \quad 0 \leq |\omega| < \pi/2K \tag{2.8}$$

While the constraints on θ_k are easily verified, the constraints on $H(z)$ can't be achieved exactly in practice. Consequently, the PQF provides only near-perfect reconstruction.

The PQF is used mainly in the MPEG-1 audio coders [ISO92]. The MPEG-1 coders use a 32-band PQF where θ_k is defined as:

$$\theta_k = \frac{\pi}{K}\left(k + \frac{1}{2}\right)\left(\frac{L-1}{2} - \frac{K}{2}\right) \tag{2.9}$$

where $K = 32$ and $L = 513$. The prototype filter is designed such that it provides more than 96dB sidelobe suppression in the stopband of each analysis channel and an output ripple less than 0.07 dB. Fig. 2.3 shows the impulse and frequency responses of the MPEG-1 prototype filter and the frequency response of the 32 MPEG-1 analysis filters.

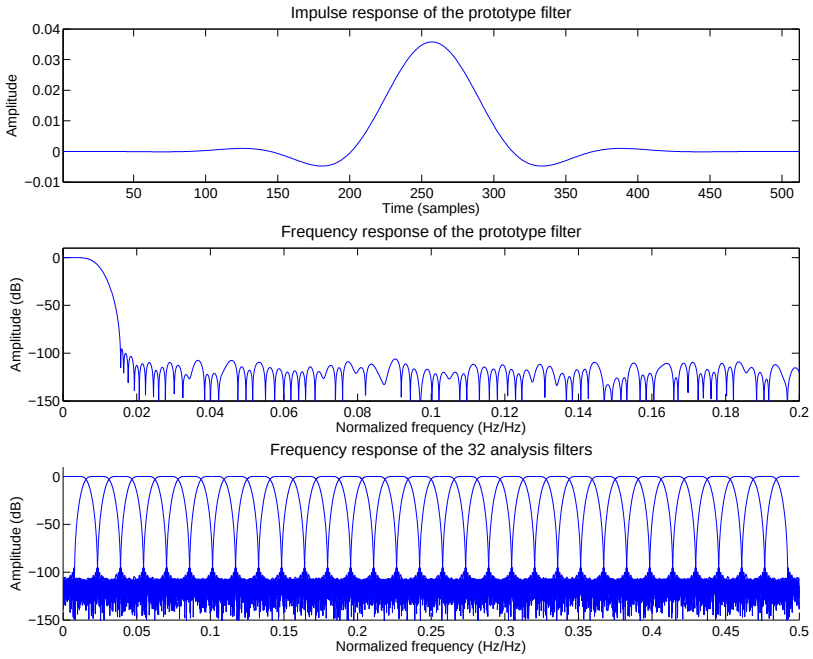

Figure 2.3: *Impulse response and frequency response of the MPEG-1 prototype filter; frequency response of the* 32 *analysis filters of the MPEG-1 PQF*

Modified Discrete Cosine Transform

The Modified Discrete Cosine Transform (MDCT) has been initially proposed by Princen, Johnson and Bradley in 1987 as the Time Domain Aliasing Cancellation (TDAC) [PB86, PJB87] and later reinvestigated by Malvar in 1990 as the Modulated Lapped Transform (MLT) [Mal90]. It is a K-band filter bank that is critically sampled and achieves perfect reconstruction. The MDCT has several advantages compared to the PQF, such as perfect reconstruction, ease of window design, good frequency resolution, ease of adapting the filterbank resolution. All these advantages have made the MDCT the filter bank of choice for most of the recent audio coders. It is used for example in MPEG-2/4 AAC [ISO98b, ISO01], Dolby AC2/3 [FBD+96] and also in MPEG-1 Layer 3 in combination with PQF [ISO92]. The filters are defined in the temporal domain as:

$$h_k(n) = h(n)cos\left[\frac{\pi}{K}\left(k+\frac{1}{2}\right)\left((L-1-n)+\frac{K}{2}+\frac{1}{2}\right)\right] \tag{2.10}$$

$$g_k(n) = h(n)cos\left[\frac{\pi}{K}\left(k+\frac{1}{2}\right)\left(n+\frac{K}{2}+\frac{1}{2}\right)\right] \tag{2.11}$$

where $h(n)$ the impulse response of the low-pass FIR prototype filter and $L = 2K$ the length of the filters. To achieve perfect reconstruction, the impulse response of the prototype filter must satisfy the following constraint

$$h^2(n) + h^2(n + L/2) = cst \text{ , with } 0 \leq n < L/2 \tag{2.12}$$

Actually, it is possible to show that the MDCT filterbank is just a particular case of the PQF where $L = 2K$ and the θ_k are defined as

$$\theta_k = -\frac{\pi}{K}\left(k+\frac{1}{2}\right)\left(\frac{L-1}{2}+\frac{K}{2}+\frac{1}{2}\right) \tag{2.13}$$

We will now reformulate the MDCT as a transform, as it has a much simpler expression. This is also the formulation we will use in this thesis. Considering a signal vector $\mathbf{x}$ of length $N = PK$, composed by P segments of length K samples each. The transform matrix $\mathbf{T}$ of size $N \times N$ corresponding to the MDCT with window length $L = 2K$ is defined as

$$\mathbf{T}(n, pK + k) = g_{p,k}(n) \text{ , } 0 \leq p < P \text{ , } 0 \leq k < K \text{ , } 0 \leq n < N \tag{2.14}$$

with

$$g_{p,k}(n) = w_p(u)\sqrt{\frac{2}{K}}\cos\left[\frac{\pi}{K}\left(u+\frac{K}{2}+\frac{1}{2}\right)\left(k+\frac{1}{2}\right)\right] \tag{2.15}$$

and

$$u = n - \left(p-\frac{1}{2}\right)K \tag{2.16}$$

and $w_p(u)$ defined on $0 \leq u < L$. The MDCT achieves perfect reconstruction if the matrix $\mathbf{T}$ is an orthogonal matrix

$$\mathbf{T}\mathbf{T}^T = \mathbf{I} \tag{2.17}$$

and this is verified if we have the following conditions:

$$\sum_{k=0}^{K-1} g_{0,k}(n)g_{0,k}(m) = \delta_{n,m} \text{ (first segment)} \quad (2.18)$$

$$\sum_{k=0}^{K-1} g_{p,k}(n)g_{p,k}(m) + \sum_{k=0}^{K-1} g_{p+1,k}(n)g_{p+1,k}(m) = \delta_{n,m} \text{ (two consecutive segments)} \quad (2.19)$$

$$\sum_{k=0}^{K-1} g_{P-1,k}(n)g_{P-1,k}(m) = \delta_{n,m} \text{ (last segment)} \quad (2.20)$$

By using the fact that

$$\sum_{k=0}^{K-1} \cos\left[\frac{\pi}{K}n\left(k+\frac{1}{2}\right)\right]\cos\left[\frac{\pi}{K}m\left(k+\frac{1}{2}\right)\right] = \frac{K}{2}\sum_{p=-\infty}^{\infty}(-1)^p(\delta_{n,m+pL}+\delta_{n,m+pL}) \quad (2.21)$$

it is easy to find the conditions on w to obtain perfect reconstruction

$$w_0(u) = 1\ ,\ 0 \le u < L/2 \quad (2.22)$$

$$w_p^2(u+L/2) + w_{p+1}^2(u) = 1\ ,\ 0 \le u < L/2\ ,\ 0 \le p < P-1 \quad (2.23)$$

$$w_{P-1}(u+L/2) = 1\ ,\ 0 \le u < L/2 \quad (2.24)$$

The two most used windows verifying the condition (2.23) are the sine window [Mal90], defined as

$$w(u) = \sin\left[\frac{\pi}{L}\left(u+\frac{1}{2}\right)\right] \quad (2.25)$$

and the Kaisel-Bessel Derived (KBD) window, proposed by the Dolby Laboratories [FBD+96].

At the analysis stage, the coefficient vector is given by $\mathbf{c} = \mathbf{T}^T\mathbf{x}$, and at the synthesis stage, the signal vector is restored using $\mathbf{x} = \mathbf{T}\mathbf{c}$. However, such matrix multiplication is never done in practice as it has a complexity of $O(N^2)$. Practical implementations use a frame-by-frame approach with an FFT-based transform (e.g. [DMP91]) which considerably reduces the complexity.

The MDCT as described previously uses a fixed window size, and thus has a fixed resolution. Fig. 2.4 shows an extract of a glockenspiel signal and the corresponding MDCT coefficients for an analysis window length of 2048 samples. In this figure, the MDCT has a long window size and thus a good frequency resolution. However it also has a poor time-resolution, the energy of an attack is spread on the entire spectrum in the 2 consecutive windows that overlap on the attack. When quantizing the coefficients of such a transform, the quantization noise of coefficients corresponding to the attacks is spread along the corresponding windows when synthesizing to the time-domain, resulting in the so-called "pre-echo artifact". To avoid this problem, a possible solution is to use a time-varying MDCT which adapts its time-frequency resolution to the signal. In state-of-the-art transform coders such as AAC, a block-switching approach is used [Edl89], where two window sizes (one long with good frequency resolution, and one short with good time resolution) are used. To achieve perfect reconstruction, it is necessary to use transitional windows between the short and long windows. Fig. 2.5 shows the results obtained with the glockenspiel signal and the time-varying MDCT used in AAC (long windows with length 2048 samples, and groups of 8 short windows with length 256 samples).

In recent work, more flexible time-varying MDCT decomposition are considered. Niamut et al. [Nia06] consider a time-varying MDCT with several window sizes and an optimal Rate-Distortion segmentation.

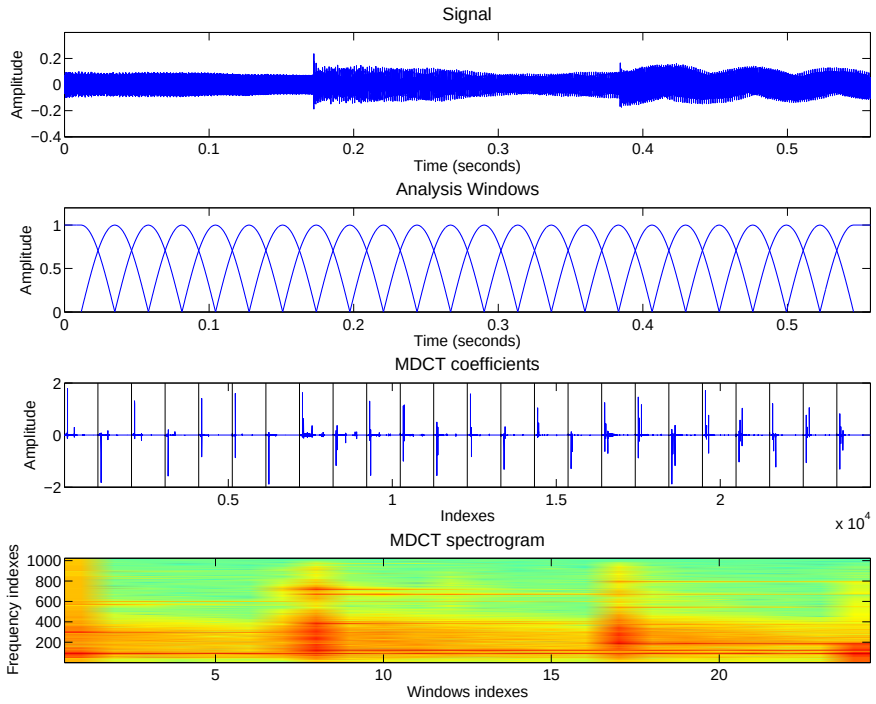

Figure 2.4: *Glockenspiel extract and MDCT with a fixed long window size (*2048 *samples)*

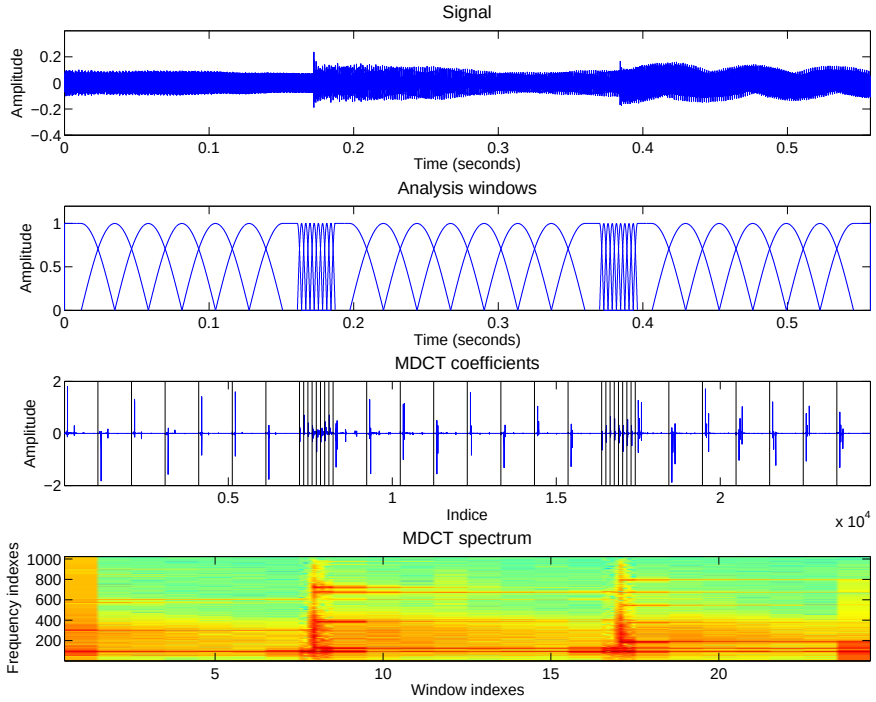

Figure 2.5: *Glockenspiel extract and time-varying MDCT with two window sizes (*2048 *and* 256 *samples)*

2.2.2 Parametric models

Filter-bank based signal representations have proved their efficiency in lossy audio codecs that provide CD-quality such as MPEG-1 Layer 3 and MPEG-2/4 AAC. However, these approaches are known to perform poorly at low bitrates. In this range of bitrates, parametric models are preferred. The basic principle of parametric modeling is to use a model of the source of the sound. While a parametric model introduces some errors even without quantization, it has the advantage that only a small number of parameters has to be encoded, facilitating efficient signal representations at low bitrates. We present here the two most used parametric models in lossy audio coding, Code Excited Linear Prediction which is an excellent model of speech signals and sinusoidal modeling which is a better model for musical signals.

Code Excited Linear Prediction

Code Excited Linear Prediction (CELP) [AS84, SA85] is a parametric representation of speech that has proven its efficiency for low bitrate speech coding. It was first proposed by Atal and Schroeder in 1984 [**?**]. CELP is based on the simple source+filter model of speech production which assumes that the vocal cords are the source of spectrally flat sound (the excitation signal), and that the vocal tract acts as a filter to spectrally shape the various sounds of speech. This model uses four main principles (see Fig. 2.6): a short term predictor, a long term predictor, a perceptual distortion measure, and a codebook of excitation signals.

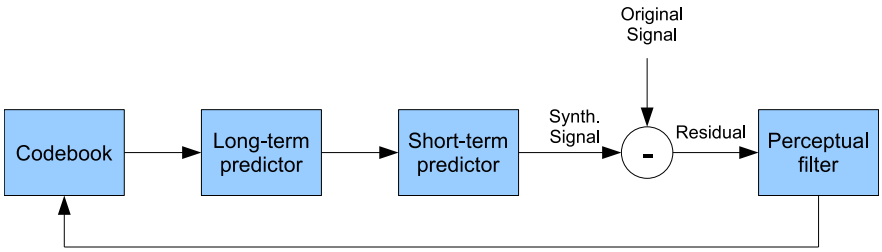

Figure 2.6: *CELP block diagram*

The short-term predictor models the spectrum shape of the signal, and the corresponding synthesis filter is defined as:

$$\frac{1}{A(z)} = \frac{1}{1 + \sum_{n=1}^{p} a_i z^{-i}} \tag{2.26}$$

with a_i are the linear prediction parameters and p the predictor order (usually between 10 and 20). The parameters a_i are estimated on a frame-by-frame basis using the standard autocorrelation technique. The long-term predictor models the pitch of the signal and the corresponding synthesis filter is defined as:

$$\frac{1}{B(z)} = \frac{1}{1 - g_p z^{-T}} \tag{2.27}$$

where T is the pitch delay and g_p is the pitch gain. These parameters are estimated on a frame-by-frame basis using an analysis-by-synthesis technique where a perceptual distortion measure is minimized. The perceptual distortion measure is the energy of the prediction error filtered by a perceptual weighting filter:

$$W(z) = \frac{A\left(z/\gamma_1\right)}{A\left(z/\gamma_2\right)} \tag{2.28}$$

This weighting filter de-emphasizes the error at the formant regions and thus exploits the psychoacoustic masking of these regions. Finally, the excitation signal is determined by selecting one innovation signal from a codebook and scale it by a gain. The best innovation signal is found using a similar analysis-by-synthesis approach as for the long-term predictor. The exhaustive search in a general codebook is very costly, and thus the design of the codebook and the search algorithm is one of the main issues of CELP coders. The solution proposed by Adoul et al. in 1987 [AMDM87, LASM90] and called Algebraic Code Excited Linear Prediction (ACELP) provides excellent performance at low computational cost and is thus used in most narrow-band speech coding standards (e.g. ETSI GSM-EFR [SL+97], ITU-T G.729 [SLA+98], 3GPP/ETSI AMR [EHJS99]).

Sinusoidal Modeling

CELP is a very good parametric model for speech signals but it is too restrictive for musical signals. A more general parametric model is sinusoidal modeling. This is a parametric representation where a signal is modeled by a sum of sinusoids with varying amplitude, frequency and phase. One of the first speech coder based on sinusoidal modeling was proposed by McAulay and Quatieri in 1986 [MQ86]. They proposed a simple model of the speech signal where it is represented by a sum of sinusoidal waves. The model parameters (amplitude, frequency and phase) are estimated from the signal spectrogram by a simple peak-picking and tracking algorithm. While this technique was originally targeted to speech signals, it has been later successfully applied to musical signals by Smith and Serra [SS87, Ser89]. Fig. 2.7 shows the model of McAulay and Quatieri applied to a trumpet signal sampled at 16 kHz.

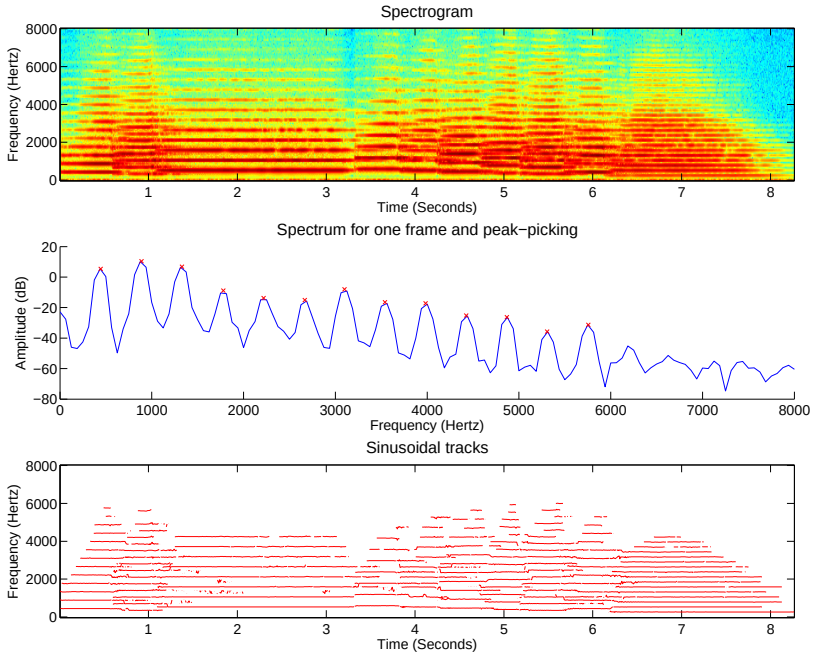

Figure 2.7: *Sinusoidal modeling algorithm of McAulay and Quatieri on a trumpet signal.*

Later in 1996, Edler, Purnhagen and Ferekidis proposed a similar sinusoidal model for audio coding [EPF96]. The main difference with the system of McAulay and Quatieri is that they use an iterative analysis-by-synthesis method [GS92] to estimate the sinusoidal parameters.

At each iteration, in the current frame, a sinusoid that best approximates the current residual is extracted and subtracted from the residual. The advantage of this approach compared to the pick-peaking approach of McAulay and Quatieri is that it reduces the false detections due to the side lobes, because when a sinusoid is subtracted from the residual, its sides lobes are removed as well. The audio coder presented in [EPF96] uses a model based on individuals sinusoids only. The same authors presented later a coder where the signal is modeled by three components [PEF98]: individual sinusoids as in [EPF96], harmonic tones composed by a sum of harmonically related sinusoids, and noise. This audio coder, called Harmonic and Individual Lines and Noise (HILN) was later standardized in the MPEG-4 framework [PM00].

The analysis-synthesis method used in HILN to extract sine waves, is also known as frame-based matching pursuit (MP) [VM99]. The MP algorithm [MZ93] is an iterative analysis-by-synthesis method that, at each iteration, selects in a dictionary the basis function that best approximates the current residual, subtracts it and iterates. Variants of Matching Pursuit use psychoacoustic masking models to extract the most perceptually relevant sinusoids. First, Verma and Meng proposed a perceptually weighted matching pursuit in 1999 [VM99], which was successfully applied to audio coding by Verma [VM00]. Second, Heusdens, Vafin and Kleijn proposed later a psychoacoustic-adaptive matching pursuit [HVK02], which was applied to audio coding by e.g. Vafin [Vaf04].

The matching-pursuit based sinusoidal modeling uses a segment-based fixed resolution method. Other methods are multi-resolution, where longer window sizes are used for estimating lower frequency sinusoids. Multi-resolution sinusoidal modeling was first introduced by Levine, Verma and Smith in 1997 [LVS97], and applied to audio coding in [LS98]. Multi-resolution sinusoidal modeling uses a filter bank approach where different analysis window lengths are used in each subband and an analysis-by-synthesis method similar to matching pursuit extract sinusoids in each subband. The sinusoids are then combined at the output of the filter bank to build sinusoidal tracks. Multi-resolution sinusoidal modeling was also considered by Oomen [Od99] and Goodwin [Goo01]. Multi-resolution sinusoidal modeling is used recently in the SinuSoidal Coder (SSC) [Od99, dSO02, SOdB03] standardized in MPEG-4 audio Amendment 2 in 2004. SSC use a three components model, where it combines multi-resolution sinusoidal modeling, with transient modeling and noise modeling.

2.2.3 Hybrid approaches

The techniques we have presented previously have all their strengths and weaknesses. On the one hand, filter-bank based approaches give best performance when used in high bitrates coding. On the other hand, parametric-model based approaches give better performance when used in low bitrates coding. We present in the following recent approaches that combine some of these techniques in order to improve the performance on a wide range of bitrates.

Linear prediction + Transform

One example of audio coding that combines linear prediction and transform based approaches is MPEG-4 Twin-VQ [IMM95], where the MDCT spectrum is flattened using the linear prediction spectrum. This allows better performance than the pure-transform AAC at low bitrates [BKS00]. Another example is 3GPP AMR-WB+ [MBB+05], where ACELP [AMDM87, LASM90] is combined with a tansform-based approach called Transform Coded eXcitation (TCX) [LSLA94]. 3GPP AMR-WB+ gives very good performance for both speech and music material at very low bitrates [MBB+05]. An approach similar to AMR-WB+ is ITU-T G.729.1 [RKT+07] that extend

a CELP coder with a MDCT-based approach in order to increase performance for general audio.

Sinusoidal Modeling + Transform

An alternative approach [vv05] combines a sinusoidal modeling based coder with a transform-based coder: SSC is used to approximate a signal and the residual is coded using a MDCT-based coder. A rate-distortion optimization is used to allocate the available bit budget among the two coders. This allows same performance as SSC at low bitrates and improved performance at high bitrates. A similar approach is found also in [VK06].

Bandwidth Extension

At very low bitrates (e.g. below 24 kbps), it is not possible to encode an audio signal without significant loss. The common approach is to encode only the lower frequencies components, which are the most perceptually relevant. This has for consequence a clean but band-limited decoded signal. An approach recently proposed by Coding Technologies and called Spectral Band Replication [DLKK02], allows the regeneration of the higher frequency components using a few parameters. This technique is used as an extension module to a core coder, allowing backwards compatibility. The core coder models only the lower half of a signal spectrum, and SBR models the higher half band by replicating the lowest band spectrum, and shaping it with additional few parameters (see Fig. 2.8). It has been successfully associated with MP3, giving an enhanced coder called mp3PRO [ZEEL02]; and also with AAC, giving a technology standardized in MPEG-4 and called HE-AAC [WKHP03]. SBR is a technique originally targeted to be used in combination with transform-based coding. Other bandwidth extension techniques have been proposed, to be used in combination with CELP-based coders, for example the techniques used in AMR-WB+ [MBB+05].

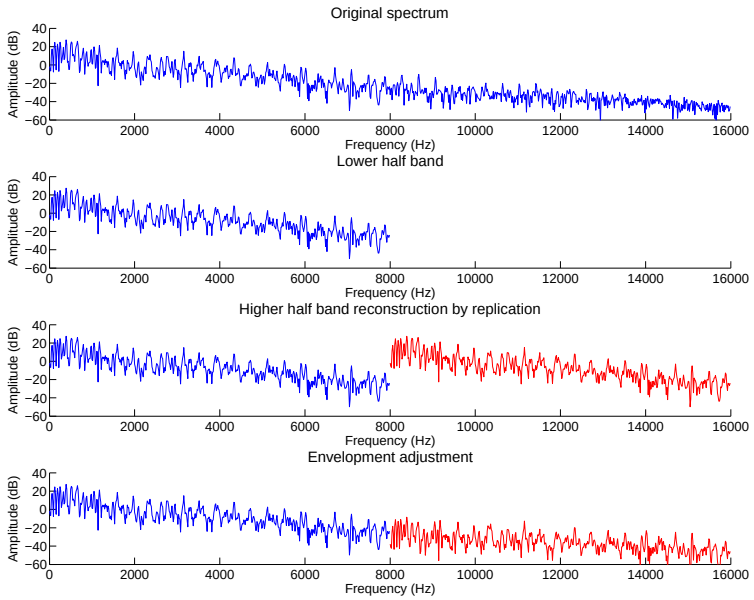

Figure 2.8: *Simplified process of Spectral Band Replication for one frame of an orchestra signal*

2.3 Source coding

We have proposed in the previous section a short review of the most popular approaches for signal representation. These techniques allow the representations of an input digital audio signal into a set of coefficients or parameters. As an example, these are the output coefficients of a MDCT-based transform, or the parameters describing a sinusoidal model. The aim of source coding is then to convert this set of coefficients or parameters into a binary format i.e. a sequence of bits and conversely convert this sequence of bits into a set of decoded coefficients or parameters. Source coding then refers to two separate operations: the coder and the decoder. These are shown in Fig. 2.9. The coder takes as input a L-length sequence of K-dimension vectors of real values and outputs a variable-length sequence of bits. In practical applications, this sequence of bits is then appended to a file or transmitted through a network. The decoder takes as input the sequence of bits produced by the coder and outputs a L-length sequence of K-dimension vectors of real values. This coding/decoding process necessarily introduces degradations i.e. the output of the decoder is not perfectly equal to the coder input, this loss is due to the first stage of the coder, called quantization, which maps the L-length sequence of K-dimension vectors of real values into a L-length sequence of integer values. In audio coding, the system is generally designed such that this loss is minimally perceived by the listener. The second stage of the coder is called entropy coding, it simply converts a sequence of integers into a sequence of bits, such that the number of bits is minimized. The decoder has the same two-stages configuration.

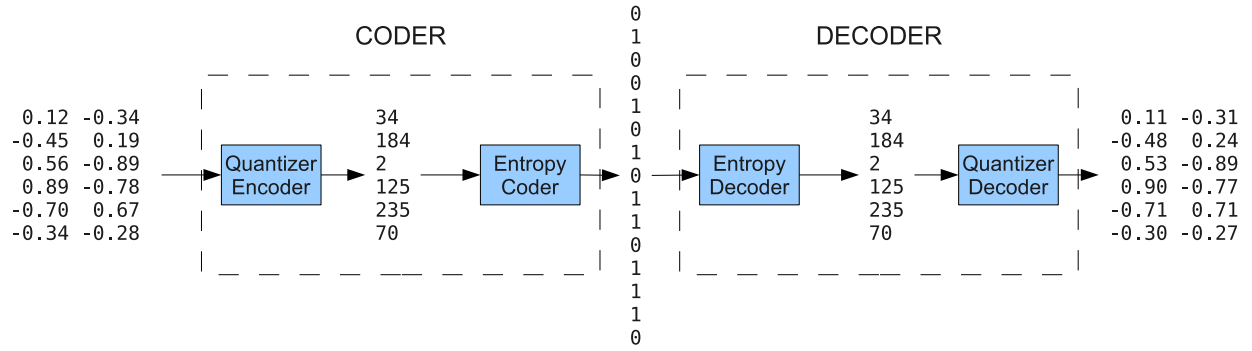

Figure 2.9: *Block diagram of lossy coding.*

In the following, we first review the quantization, then the lossless coding, and finally introduce a particular source coding technique called bitplane coding.

2.3.1 Quantization

Computers cannot deal with real values (i.e. with infinite precision), the goal of quantization is then to approximate real values using integer values that are easily processed by computers. The quantization operation in the coder is called "quantizer encoder" and is defined as

$$\alpha : \mathbb{R}^K \rightarrow \mathcal{I} = \{0, 1, ..., N-1\} \tag{2.29}$$

where α is the quantizer encoder with dimension K and size N. The quantizer encoder is the only non-reversible and thus lossy operation in the whole process. Associated with the quantizer encoder is a partition of $\mathbb{R}^K$ into N cells $\mathbf{R}_i$ for all $i \in \mathcal{I}$, defined as

$$\mathbf{R}_i = \{\mathbf{x} \in \mathbb{R}^K : \alpha(\mathbf{x}) = i\} \tag{2.30}$$

In the decoder, the corresponding quantization operation is called "quantizer decoder" and is defined as

$$\beta : \mathcal{I} \rightarrow \mathcal{C} = \{\mathbf{y}_0, \mathbf{y}_1, ..., \mathbf{y}_{N-1}\} \tag{2.31}$$

where β is the quantizer decoder with dimension K and size N, and $\mathbf{y}_i \in \mathbb{R}^K$ for all $i \in \mathcal{I}$ are the output points (see examples of 2-D quantizers with cells and output points in Fig. 2.11). Quantization then refers to the cascade of the quantizer encoder and quantizer decoder

$$\mathcal{Q} : \mathbb{R}^K \rightarrow \mathcal{C} \tag{2.32}$$

and $\mathcal{Q}$ is then called the quantizer.

The quantizer input is generally assumed to be a random variable X with probability density function (pdf) $f_{\mathbf{X}}(\mathbf{x})$. The overall performance of a quantizer is then generally characterized by its average distortion and average rate. The average distortion is defined as

$$D = \sum_{i=0}^{N-1} \int_{\mathbf{R}_i} d(\mathbf{x}, \mathbf{y}_i) f_{\mathbf{X}}(\mathbf{x}) d\mathbf{x} \tag{2.33}$$

where $d(.,.)$ is a distortion measure, which can be as simple as the squared-error distortion or more complicated pyschoacoustically-motivated distortion measures. The common approach is to use a psychocoustic model that is able to measure how well the degradations introduced by the quantizer will be perceived by the human auditory system (good reviews on psychoacoustic models for audio coding are [PS00] and [BG03]). The average rate corresponds to the average number of bits needed to described the quantizer encoder output, and it is defined as

$$R_{\text{r.c.}} = \log_2 (N) \tag{2.34}$$

if all $i \in \mathcal{I}$ are described with an equal number of bits, a case called "resolution-constrained". Otherwise, it is refered as the "entropy-constrained" case, and the lowest possible average rate is then defined as the entropy of the quantizer encoder output

$$R_{\text{e.c.,min}} = -\sum_{i=0}^{N-1} \int_{\mathbf{R}_i} f_{\mathbf{X}}(\mathbf{x}) d\mathbf{x} \log_2 \left(\int_{\mathbf{R}_i} f_{\mathbf{X}}(\mathbf{x}) d\mathbf{x} \right) \tag{2.35}$$

The design of a quantizer is then to optimize the rate-distortion tradeoff, which can be done in several ways: either by optimizing distortion for a constrained rate, by optimizing rate for a constrained distortion, or by an unconstrained optimization using a Lagrange approach. Good reviews on the design of optimal quantizers are [GG92] and [GN98]. Below, we introduce briefly the most used quantizers in lossy audio codecs.

If $K = 1$, the input of the quantizer is a scalar, and this particular case is then called "scalar quantization". This is the approach traditionally chosen in many audio coders, due to its low computational complexity and storage costs. The simplest scalar quantizer is the "uniform quantizer", defined as equal-size cells with centered output points. A symmetric uniform quantizer with a number of cells N even is also known as "midrise quantizer" and with N odd as "midtread quantizer" (see Fig. 2.10). Most recent audio codecs are based on non-uniform quantization, for example in MPEG-1 Layer 3 [ISO92] and MPEG-2/4 AAC [ISO98b, ISO01] to quantize the MDCT coefficients. The non-uniform quantizer of MP3 and AAC raises its input to the 3/4 power before applying a midtread quantizer (see Fig. 2.10); this way, bigger values are quantized with less accuracy than smaller values. Moreover, the midtread quantizer used in MP3 and AAC has generally a bigger central cell, a case also called dead-zone quantizer in the literature.

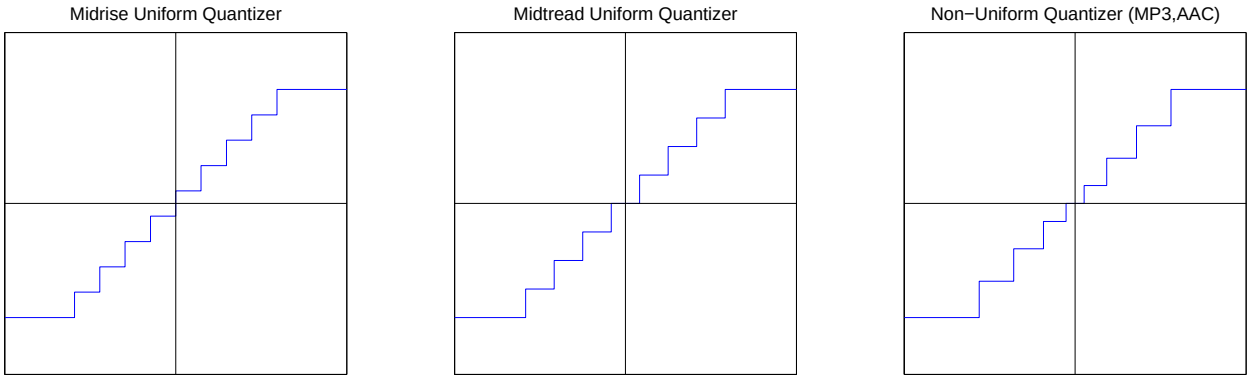

Figure 2.10: *Input/Output functions for several scalar quantizers. From left to right: a midrise uniform quantizer (with $N = 10$); a midtread uniform quantizer (with $N = 9$); the non-uniform quantizer used in MP3,AAC (with $N = 9$).*

Vector quantization is the generalization of scalar quantization to higher dimension $K > 1$. Vector quantization allows better performance than scalar quantization but at higher computational complexity and storage costs. It is used for example to quantize the flattened MDCT coefficients in Twin-VQ [IMM95], or to quantize the parameters of a CELP model (e.g. [SL$^+$97, SLA$^+$98, EHJS99]). A good reference on vector quantization is [GG92]. Particular cases of vector quantization are found in audio coding that use sinusoidal modeling. The first particular case is called polar quantization and it is the natural 2-dimension quantizer for circularly symmetric data. It is used e.g. in [VK05] to quantize the amplitude and phase of sinusoids. The extension of polar quantization to 3 dimensions is used to quantize the amplitude, the phase and the frequency of sinusoids in e.g. [VPK05], it is also known as spherical quantization in e.g. [KJH04]. Fig. 2.11 shows examples of 2-dimension vector quantizers: a general vector quantizer, a quantizer obtained by using two scalar quantizers and called rectangular quantizer, and a polar quantizer.

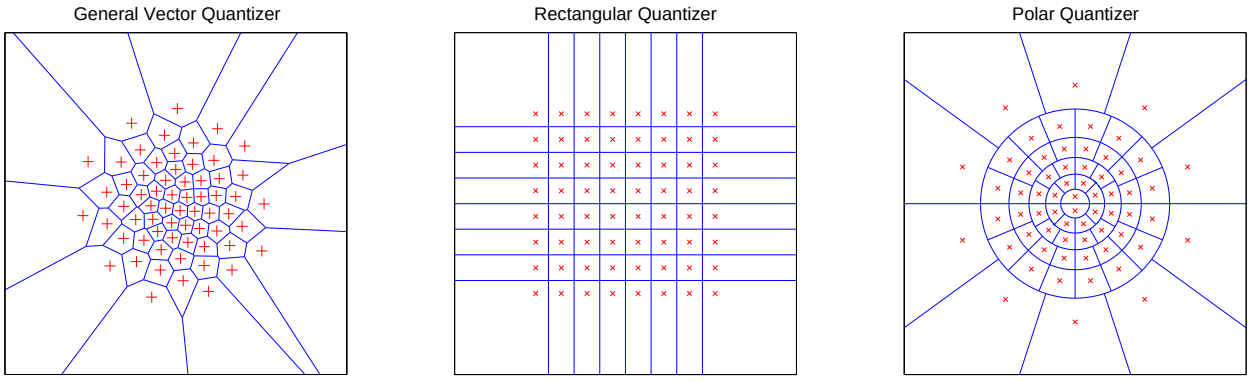

Figure 2.11: *Examples of 2-dimension vector quantizers with size $N = 64$. From left to right: a general vector quantizer; a rectangular quantizer; a polar quantizer.*

2.3.2 Entropy coding

Entropy coding aims at converting a sequence of integers into a sequence of bits such that the average number of bits is minimized. It is defined as a reversible function γ from $\mathcal{I}^L$ to $\mathcal{J}$, where $\mathcal{I} = \{0, 1, ..., N-1\}$ and $\mathcal{J}$ is a collection of variable-length binary vectors. The input is generally assumed to be a sequence of L realizations of a i.i.d. random variable with known probabilities.

The most widely used entropy coding technique is Huffman coding [Huf52]. The basic principle of Huffman coding is to associate to each integer in $\mathcal{I}$ a variable length binary vector in such a way that shorter binary vectors correspond to more probable input integers. For a L-length input sequence, Huffman coding then simply concatenates the binary vectors associated to each integer. The algorithm produces prefix codes, it means that the bit string representing a binary vector is never a prefix of the bit string representing any other binary vector: this property allows the decoder to distinguish binary vectors of different integers in the bitstream. Huffman coding is the optimal entropy coding technique that maps a finite set of integers to a set of variable-length binary vectors with the prefix condition, where the input probabilities are known. It is used for example in MP3 [ISO92] and AAC [ISO01] to code the quantizer encoder output.

Another, yet less known, entropy coding technique is Golomb coding [Gol66]. It is similar to Huffman coding as it associates to each integer in $\mathcal{I} = \{0, 1, ..., N-1\}$ a unique variable-length code that has the prefix condition. However, Golomb coding uses a much simpler and more restrictive probability model than Huffman coding, where small input values are assumed to have a higher probability than large input values. On the other hand, Golomb coding is able to deal with infinite N, where Huffman coding cannot. It has been shown that Golomb coding is optimal for an input with geometric distribution; this is then useful for run-length encoding of a sequence of bits where the run-lengths follow a geometric distribution.

Arithmetic coding [Pas76, Ris76, WNC87, Sai04] is different from Huffman and Golomb coding as there is no exact relationship between an integer and a variable-length binary vector, a binary vector is associated to the entire input sequence of integers. Arithmetic coding is the optimal source coding technique for infinite-length sequence of integers input, it is used for example in MPEG-4 BSAC [PKS97] and MPEG-4 SLS [YRXK06]. See [Sai04] for a good report on arithmetic coding.

The algorithms described above assume that the input is i.i.d. and that the probabilities are known. In situations where this is not the case, adaptive versions of these algorthims are preferred (e.g. adaptive Huffman coding [Gal78, Vit87], adaptive Golomb coding [Lan83] and adaptive Arithmetic coding [WNC87, Sai04]).

2.3.3 Bitplane coding

Bitplane coding is an efficient combination of embedded quantization and lossless coding that allows the coding of a sequence of real values in an embedded manner. Historically, bitplane encoding was first used in wavelet-based image coding [Sha93, SP96], and only later in MDCT-based audio coding [PKS97, RMB02, Dun06]. In the following, we first introduce embedded quantization, then we describe the basic bitplane coding algorithm.

Embedded quantization refers to sets of quantizers that allow the successive refinement of an input vector. In most bitplane coding approaches, the embedded quantizers are simple scalar dead-zone quantizers similar to a midtread quantizer whose central cell width is double. We detail the expression of these simple embedded quantizers in the following. We assume in the following that the input is bounded and that the quantizers have a fixed cell width, but this is easily extended to variable cell width by scaling the quantizers input. Considering P scalar quantizers

$\mathcal{Q}^{(p)}, p = 1, ..., P$ with size $N^{(p)} = 2^{p+1} - 1$, the output points of the quantizer $\mathcal{Q}^{(p)}$ are defined as

$$y_i = (i - 2^p + 1/2)2^{-p} \ , \ i = 0, 1, .., 2^p - 2 \tag{2.36}$$
$$y_{2^p-1} = 0 \tag{2.37}$$
$$y_i = (i - 2^p + 3/2)2^{-p} \ , \ i = 2^p, 2^p + 1..., 2^{p+1} - 2 \tag{2.38}$$

and the cells are defined as

$$\mathbf{R}_0 = \{x \in \mathbb{R} : x < b_0^{(p)}\} \tag{2.39}$$
$$\mathbf{R}_i = \{x \in \mathbb{R} : b_{i-1}^{(p)} \leq x < b_i^{(p)}\} \ , \ i = 1, .., N^{(p)} - 2 \tag{2.40}$$
$$\mathbf{R}_{N^{(p)}-1} = \{x \in \mathbb{R} : b_{N^{(p)}-2}^{(p)} \leq x\} \tag{2.41}$$

with $b_i^{(p)}, i = 0, ..., N - 2$ are the cell boundaries defined as

$$b_i^{(p)} = (i - 2^p + 1)2^{-p} \ , \ i = 0, 1, .., 2^p - 2 \tag{2.42}$$
$$b_i^{(p)} = (i - 2^p + 2)2^{-p} \ , \ i = 2^p, 2^p + 1..., 2^{p+1} - 2 \tag{2.43}$$
$$\tag{2.44}$$

Figure 2.12 shows the first four quantizers $\mathcal{Q}^{(p)}, p = 1, ..., 4$.

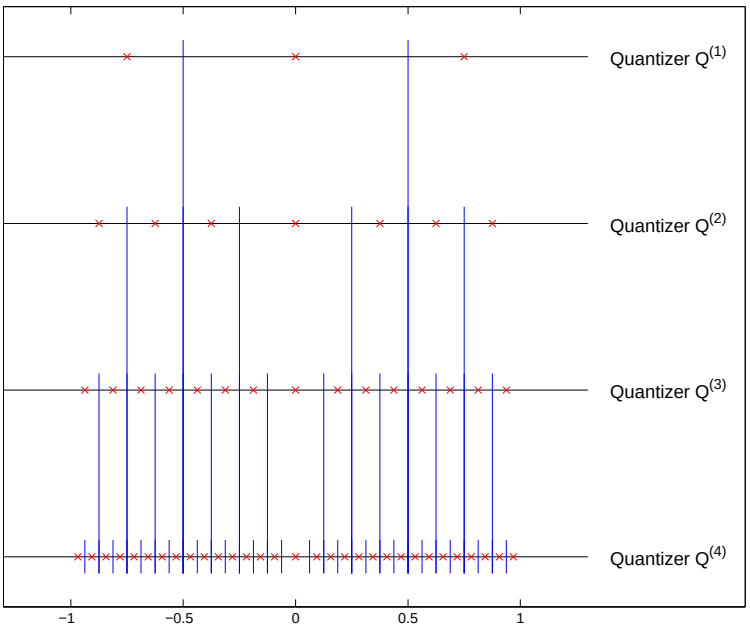

Figure 2.12: *Example of simple scalar embedded quantizers (the cross are the output points and the vertical lines are the cell boundaries).*

Considering a real value x, the output of the quantizer encoder corresponding to $\mathcal{Q}^{(p)}$ is then represented by a binary vector $c_0c_1...c_{p+1}$, where the first bit c_0 represents the sign of x, and the following bits $c_1...c_{p+1}$ represent the successive refinements of the amplitude of x

$$c_j = \text{mod}(\text{floor}(\text{abs}(x) * 2^j), 2), \ j = 1, ..., p + 1 \tag{2.45}$$

Considering a sequence of real values $x_0, x_2, ..., x_{L-1}$, the quantizer encoder corresponding to $\mathcal{Q}^{(p)}$ produces a binary vector per input which are then concatenate in order to produce a binary matrix (as shown in Fig. 2.13 for $P = 4$ and $L = 12$), where the column l corresponds to the quantizer encoder output of x_l, and the row j (also called level j) is called the j-th bitplane noted B_j. An input x_l is said to be significant at level j if $\text{abs}(x_l) \geq 2^{-j}$. The significance of each input is stored in a vector $\mathbf{z}$ ($z_l = 1$ if the coefficient is significant, $z_l = 0$ if not).

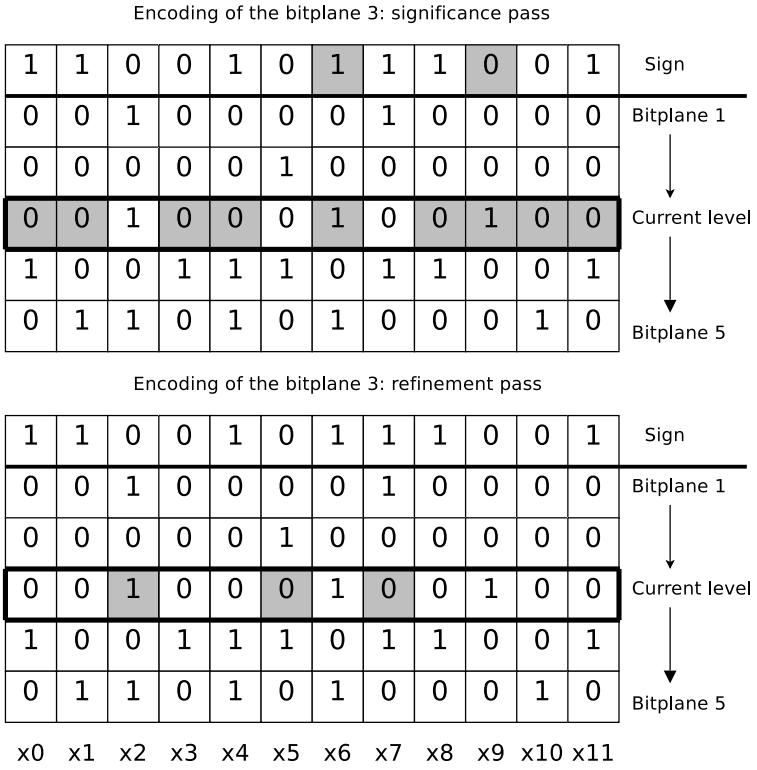

Figure 2.13: *One iteration of the simple bitplane encoder. The two encoding passes are shown. The bits in gray corresponds to the transmitted bits.*

The basic principle of bitplane encoding is to send successively each bitplane starting from the first bitplane. This is generally done using a scheme in two passes: the significance pass and the refinement pass. The significant pass transmits the subset of the bitplane B_j corresponding to the bits of the non already significant coefficients $BS_j = \{c_{j,l} | z_l = 0\}$. The significance pass also transmits the sign of the new significant coefficients. The refinement pass transmits the subset of the bitplane B_j corresponding to the bits of the already significant coefficients $BR_j = \{b_{j,l} | z_l = 1\}$. All existing bitplane encoding algorithms differ essentially in the way they perform the significance pass. The refinement pass is generally performed simply by sending directly the bits of BR_j. Image coders such as EZW [Sha93] and SPIHT [SP96] use a bitplane encoding algorithm with a significance pass that uses the tree structure inherent of the wavelet transform. Though this tree structure is less evident for a MDCT transform, similar bitplane encoding algorithms as SPIHT have been applied to MDCT-based audio coding [RMB02, RMB03, SZM05]. Another way to perform the significance pass is to use arithmetic coding to code the bits in BS; this is the technique chosen by the MPEG-4 standards BSAC [PKS97] and SLS [YRXK06]. Finally, a technique proposed in [Dun06] encode the bits in BS using Golomb runlength encoding.

2.4 Evaluation

Considering a PCM input signal $\mathbf{x}$ of length N called "reference signal", a lossy audio coder produces another signal $\hat{\mathbf{x}}$ called "test signal". The aim of audio coding is to obtain a test signal with a minimum of degradations in the perceived audio quality as compared to that of the reference signal. It is thus necessary to evaluate these degradations from a perceptual point of view. As the human listeners are the ultimate judges of quality, formal listening tests are considered as the only satisfactory way to evaluate the performance of an audio coder. However, listening tests are very time consuming as they have to involve a consequent number of trained "expert" listeners in order to obtain relevant results. This is particularly true when one has to compare many signals and coder configurations. Consequently, recent research has focus on trying to find efficient objective measures that correlate well with listening tests results. Though these objective measures are far from being perfect, they give a good indication on the perceived audio quality and they are useful e.g. for tuning the coder parameters. We present in the following the most used listening tests methods, and some of the most recent objective measures.

2.4.1 Listening tests

ITU-R BS.1116-1 It is referred to as "Methods for the subjective assessment of small impairments in audio systems including multichannel sound systems" [ITU97]. It is also called ABC/HR, or just ABC. This test is generally used for the evaluation of (near-)transparent audio coding. The BS.1116-1 test gives to the subject three stimuli to listen to. These are the reference signal, a signal A and a signal B. Either A or B is the same as the reference signal, and is thus called hidden reference. The other one is the test signal. The subject has to give a score to both signals A and B, on a 5-point scale (1. Very annoying, 2. Annoying, 3. Slightly annoying, 4. Perceptible, but not annoying and 5. Imperceptible). The subject is forced to give the maximum score to at least one of the two signals. The final score is the difference between the test signal score and the reference signal score, it is called Subjective Difference Grade (SDG). As an example, this test has been used by MPEG for evaluating MPEG-2 AAC at high bitrates [ISO98a, ISO96].

ITU-R BS.1534-1 It is referred as "Method for the subjective assessment of intermediate quality levels of coding systems" [ITU03], but it is usually called MUltiple Stimuli with Hidden Reference and Anchors (MUSHRA). This test is usually used to evaluate medium to high quality audio coding. The BS.1534-1 test gives the subject several stimuli to hear. These include the reference, an hidden reference, one or more anchors and several tests signals. The anchors are generally low-pass versions of the reference signal, and aim at providing a low-quality signal that scale the output such that test signals are not scored too low. The subject is asked to evaluate each stimuli using scores that range from 0 (extremely poor quality) to 100 (transparent). As an example, this test has been used by MPEG for evaluating MPEG-4 HE-AAC [ISO03] and MPEG-4 SSC [ISO04b]. Several open-source implementations of the evaluation interface are available (e.g. MUSHRAM [Vin08], see Fig. 2.14).

2.4.2 Objective Measures

The simplest objective measure is the Signal to Noise Ratio (SNR), defined as

$$SNR(\mathbf{x}, \hat{\mathbf{x}}) = 10 \log_{10} \left(\frac{\sum_{n=1}^{N} x_n^2}{\sum_{n=1}^{N} (x_n - \hat{x_n})^2} \right) \tag{2.46}$$

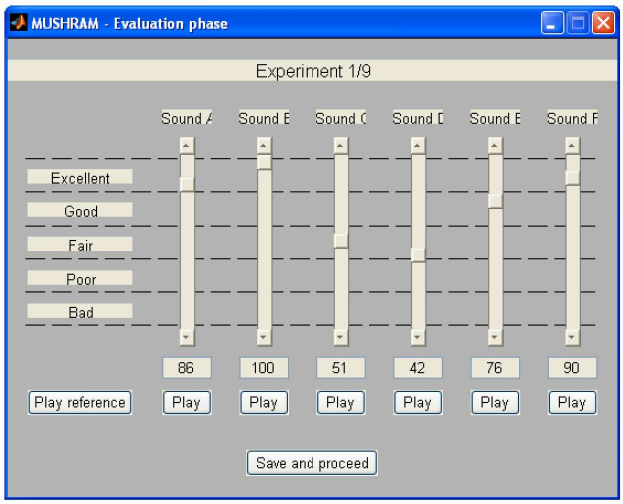

Figure 2.14: *MUSHRAM: A matlab interface for B.1534-1.*

However, it is obvious that SNR is far from being a good measure of the perceived quality of coded audio. One well known example that illustrates this purpose is the comparison of two tests signals, one obtained by passing a reference signal through a high quality audio coder (e.g. AAC), another one by adding white noise to the same reference signal such that the resulting SNR is the same as the SNR of the first test signal. In most cases, one cannot distinguish the reference signal from the first test signal, whereas the second test signal is easily distinguished. More complex measures are thus generally used (Fig. 2.15 shows a simplified block diagram of an objective measure). These approaches are based on an auditory model, which is applied to the reference and tests signal. These auditory models produce an internal representation for each signal, which are then compared in order to produce an objective measure. This measure ideally scales with the perceived signal differences. In the following, we present two of the most recent existing objective measures.

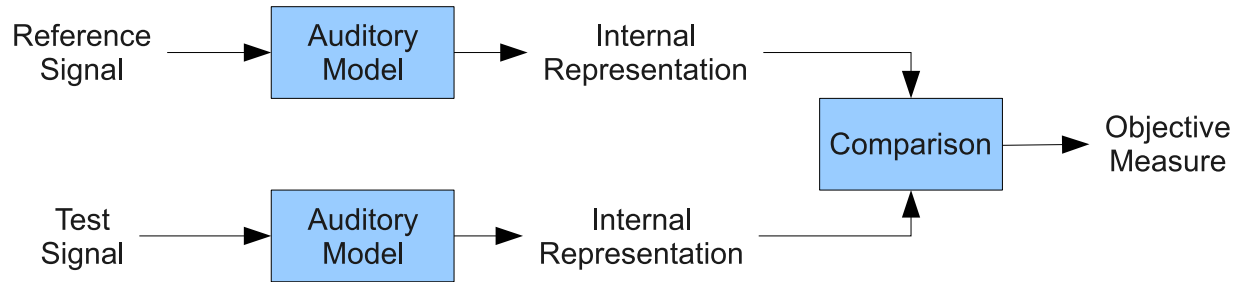

Figure 2.15: *Simplified block diagram of an objective measure.*

PEAQ Perceptual Evaluation Audio Quality [TTB+00] is the only standardized objective measure for the assessment of audio coding quality. It has been standardized by ITU-T as BS.1387 in 1998 [ITU98] and revised as BS.1387-1 in 2001 [ITU01]. PEAQ is a combination of several auditory models developed independently. Several features are extracted from these models for the reference and the test signals and are passed trough a neural network. This neural network outputs an objec-

tive measure on the same scale than the Subjective Difference Grade used in BS.1116, and is thus called Objective Difference Grade (ODG). The neural network is trained using a large database of listening tests performed on various signals and coders, using the BS.1116 method. It is thus very specialized to the task of evaluating small impairment in audio coding systems. A commercial implementation of PEAQ is available in a proprietary software called Opera (OPTICOM [Opt08]). The PEAQ standard has also been well studied by P. Kabal in [Kab03], and his team has implemented an open source version of PEAQ but he claims in [Kab03] that the standard described in [ITU98] and [ITU01] is not detailed enough for a conforming implementation.

PEMO-Q PEMO-Q has been proposed by Huber and Kollmeier in 2006 [HK06] and it is an extension of an objective measure of speech quality proposed by Hansen and Kollmeier in 2000 [HK00]. PEMO-Q uses an auditory model based on the work of Dau et al. [DPK96, DKK97] to compute internal representations of the reference and the test signals. Then the cross-correlation of these representations is computed and produces a first objective measure called Perceptual Similarity Measure (PSM), whose value range from 0 (extremely poor quality) to 1 (transparent). PEMO-Q also produces a second objective measure, which is the fifth percentile of the sequence of instantaneous PSM. This second measure is mapped to the Subjective Difference Grade used in BS.1116 and is thus called Objective Difference Grade, as PEAQ. The main difference with PEAQ is that PEMO-Q does not use a machine learning system (neural network in PEAQ) trained with a large database of listening tests results, and thus is less specialized than PEAQ. Authors claim that PEMO-Q is able to predict not only the perceived quality of coded audio, but of any distorted audio signals.

2.5 Conclusions

We have proposed in this first chapter a short review of lossy audio coding. We have first introduced the main signal representation approaches used in state-of-the-art lossy audio codecs. We have seen that there are 3 main categories: the filter banks, the parametric models, and the hybrid approaches. The filter banks are preferred for CD-quality audio codecs such as MP3 and AAC, while parametric modeling give better performance in low bitrates speech and audio codecs such as AMR and SSC. In this thesis, we consider CD-quality audio coding, and investigate new signal representations that perform better that the filter banks (particularly the state-of-the-art MDCT) at low bitrates. These new signal representation approaches are based on a class of signal representations known as sparse signal representations. This is a research domain that has also found interests in other coding applications such as image and video coding. We will review sparse signal representations in the next chapter.

In the second section, we have then reviewed source coding. We have briefly introduced quantization and entropy coding, and we have particularly emphasized a source coding technique called bitplane coding that allows scalable coding. This technique was originally proposed for image coding, but is now used also in scalable audio codecs such as the standards MPEG-4 BSAC and MPEG-4 SLS.

Finally, we have introduced several techniques for evaluating lossy audio coding. Both listening tests and objective evaluation measure will be used in this thesis for evaluating the proposed lossy audio codecs.

Chapter 3

Sparse signal representations

Abstract

In this chapter, we review the foundations of the so-called sparse representations, that will be used as the primary tool for our new audio codec. As there are numerous approaches in this research domain, we focus on the techniques related to our study, namely the greedy algorithms (e.g. Matching Pursuit) and the dictionaries used in the coding applications.

Contents

3.1 Introduction

Sparse signal representations aim at representing a signal with a small number of elementary functions. These functions are chosen among a collection of arbitrary size, which may be much larger than the signal length. Sparse signal representations find obvious applications in coding such as audio coding (e.g. [Vaf04, VP07]), image coding (e.g. [FiVVF06]) and video coding (e.g. [NZ97, GMPV04]). They have also been applied successfully to numerous other applications, examples of recent work include: image decomposition [SED05]; blind source separation [FG06]; convolutive blind source separation [HXDC07]; multimodal signal analysis [MJV+07]; image denoising, inpainting and demosaicing [MMES08]; audio denoising [FTDG08]; musical instrument recognition [LLV+08].

The first main problem of sparse representations is to find "the best representation". This actually implies two issues, how to define the best representation, and how to find this representation. This will be discussed in the sections 3.2 and 3.3. The second main problem is to choose the collection of elementary functions that will be used to represent a signal. In most applications, the elementary functions are deterministic waveforms. We review some of these waveforms in section 3.4. As we are interested in this thesis in coding applications, we restrain our review to the sparse representations used in coding applications (audio, image and video).

3.2 Problems and solutions

3.2.1 Mathematical settings

We consider a column vector (the signal) $\mathbf{x}$ in $\mathcal{R}^N$. The signal $\mathbf{x}$ is approximated (with possible equality) by a linear combination of K unit-norm column vectors $\mathbf{g}_k$

$$\mathbf{x} \simeq \sum_{k=1}^{K} c_k \mathbf{g}_k \tag{3.1}$$

The vectors $\mathbf{g}_k$ belong to $\mathcal{R}^N$ and are called "atoms" and the scalars c_k are the coefficients. The finite collection of all atoms is called "dictionary" and is written as

$$\mathcal{D} = \{\mathbf{g}_k, 1 \leq k \leq K\} \tag{3.2}$$

To simplify notations, we define the synthesis matrix $\mathbf{\Phi}$ of size $N \times K$, where the columns $\mathbf{\Phi}_k$ of $\mathbf{\Phi}$ correspond to the atoms $\mathbf{g}_k$; $\mathbf{\Phi}_\Gamma$ the submatrix of $\mathbf{\Phi}$ containing only those columns of $\mathbf{\Phi}$ with indices in a set of indices Γ; the column vector $\mathbf{c}$, where elements are the coefficients c_k; and $\mathbf{c}_\Gamma$ is the subvector of $\mathbf{c}$ containing only those elements of $\mathbf{c}$ with indices in Γ. We then have

$$\mathbf{x} \simeq \mathbf{\Phi c} \tag{3.3}$$

If there is equality in (3.3), the signal is represented exactly by a linear combination of atoms, and the vector of coefficients $\mathbf{c}$ is then called "exact representation" (or just representation). Otherwise, the signal is approximated with more or less precision by a linear combination of atoms, the vector of coefficients $\mathbf{c}$ is then called "approximation", and $\hat{\mathbf{x}} = \mathbf{\Phi c}$ is called the "approximant" of $\mathbf{x}$. We consider these two cases in the following.

3.2.2 Exact signal representations

We consider first the case of an exact signal representation, the signal is then represented exactly by a linear combination of atoms

$$\mathbf{x} = \mathbf{\Phi c} \tag{3.4}$$

The problem is now to find a representation $\mathbf{c}$, given a signal $\mathbf{x}$ and a dictionary $\mathbf{\Phi}$. We distinguish two cases in the following. Firstly, if the dictionary is an orthonormal basis, then the solution is straightforward. Secondly, if the dictionary is overcomplete, then the problem is generally more complicated.

Orthonormal basis

We first consider the simple case of the orthonormal basis. The dictionary is an orthonormal basis if $K = N$ and the atoms are mutually orthogonal. In this case, $\mathbf{\Phi}$ is an orthogonal matrix and $\mathbf{\Phi}$ is invertible with $\mathbf{\Phi}^{-1} = \mathbf{\Phi}^T$. The signal representation is then unique and is given by

$$\mathbf{c} = \mathbf{\Phi}^T \mathbf{x} \tag{3.5}$$

Overcomplete dictionary

Sparse representations The dictionary is overcomplete when it spans the entire signal space $\mathcal{R}^N$ and its size is superior to the signal dimension $K > N$. In this case, the signal representation is not unique. There exists an infinite number of representations that verify (3.4). In this chapter, we are interested in sparse representations, which are the representations, among all possible representations, that concentrate the signal energy on few coefficients. Sparse representations are then the representations that minimize a measure of "sparseness". Several measures of the "sparseness" of a representation have been proposed in the literature. The most popular measure is the "p-like-norm sparseness measure" $E^{(p)}(\mathbf{c})$ with $0 \leq p \leq 1$ defined as

$$E^{(p)}(\mathbf{c}) = \left(\sum_{k=1}^{K} |c_k|^p \right)^{\frac{1}{p}} \tag{3.6}$$

for $p > 0$ and

$$E^{(0)}(\mathbf{c}) = \lim_{p \to 0} \sum_{k=1}^{K} |c_k|^p \tag{3.7}$$

Using this sparseness measure, a sparse representation is then defined as a solution of the following problem

$$P_p : \min_{\mathbf{c}} E^{(p)}(\mathbf{c}) \text{ subject to } \mathbf{x} = \mathbf{\Phi c} \tag{3.8}$$

Ideal solution The 0-like-norm measure is often used to define the sparsest possible representation, as $E^{(0)}(\mathbf{c})$ is equal to the number of non-zero terms in $\mathbf{c}$. However, Natarajan [Nat95] has shown that solving the problem P_0 is NP-hard if the dictionary is unrestricted, it means that it is solved only by an exhaustive search among all possible combinations of atoms. In practical applications, such an exhaustive search is clearly impossible.

Method of frames A straightforward solution exists for the generalized case $p = 2$. Indeed, the generalized problem P_2 corresponds to the least square problem

$$P_2 : \min_{\mathbf{c}} \sum_{k=1}^{K} c_k^2 \text{ subject to } \mathbf{x} = \mathbf{\Phi c} \tag{3.9}$$

which has a unique solution given by

$$\mathbf{c}^{opt} = \mathbf{\Phi}^{+}\mathbf{x} \tag{3.10}$$

where $\mathbf{\Phi}^{+}$ is the pseudo-inverse of $\mathbf{\Phi}$ and is defined as

$$\mathbf{\Phi}^{+} = \mathbf{\Phi}^{T}\left(\mathbf{\Phi}\mathbf{\Phi}^{T}\right)^{-1} \tag{3.11}$$

This is the solution adopted by the method of frames [Dau88]. The solution given by P_2 has minimum energy but this energy is generally spread on many coefficients and thus is generally not sparse [CDS99].

Basis Pursuit Sparse representations are obtained when considering the case $0 < p \leq 1$. Chen, Donoho and Saunders considered the case $p = 1$ in [CDS99]. The corresponding problem, called Basis Pursuit, is

$$P_1 : \min_{\mathbf{c}} \sum_{k=1}^{K} |c_k| \text{ subject to } \mathbf{x} = \mathbf{\Phi c} \tag{3.12}$$

It is possible to formulate this problem as a Linear Programming problem and thus solve it by standard Linear Programming algorithms, such as the simplex method or the interior point method.

FOCUSS Rao and Kreutz-Delgado considered the more general case $0 < p \leq 1$ in [RKD99] and proposed a class of algorithms called FOCal Undetermined System Solver (FOCUSS) for the problem

$$P_{0<p\leq 1} : \min_{\mathbf{c}} \sum_{k=1}^{K} |c_k|^p \text{ subject to } \mathbf{x} = \mathbf{\Phi c} \tag{3.13}$$

The authors also considered other sparseness measures in [RKD99], such as the Gaussian entropy and the Shannon entropy.

Best Orthogonal Basis The Shannon entropy has also been considered by Coifman and Wickerhauser in [CCW92]. They have proposed a method, called Best Orthogonal Basis, that selects the best orthogonal basis among a union of orthogonal basis, the best one being the one that minimizes the entropy of the representation. However, this algorithm has severe limitations: the dictionary is restricted to a union of orthonormal basis, and the signal cannot be represented by elements from different basis.

3.2.3 Signal approximations

Sparse approximations We now consider the case of a signal approximation. The signal is then approximated by a linear combination of atoms

$$\mathbf{x} \simeq \hat{\mathbf{x}} = \mathbf{\Phi c} \tag{3.14}$$

We are interested here in sparse approximations. These are defined as approximations whose approximant verifies the following two properties: its energy is concentrated on few coefficients, and it is as close as possible to the signal. The measure of the approximation error is generally based on the l_2 norm, and is given by

$$\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 = \|\mathbf{x} - \mathbf{\Phi c}\|_2^2 \tag{3.15}$$

but it is important to remark that other approximation measures are also used in specific applications (e.g. psychoacoustic-motivated distortion measure in parametric audio coding [HVK02, VP07]). To measure the sparseness of the approximation, as for the exact representation case, the p-like-norm is a popular choice. A sparse approximation is thus generally defined as the solution of the following problem

$$P_{p,\lambda} : \min_{\mathbf{c}} \|\mathbf{x} - \mathbf{\Phi c}\|_2^2 + \lambda E^{(p)}(\mathbf{c}) \tag{3.16}$$

The parameter λ balances the two terms of the sum. If λ is large, then the solution is very sparse and the error is large; if λ is small, the error is small but the solution is less sparse.

Ideal solution The l_0-norm-like is generally used to define the sparsest possible approximation, but as for the exact representation problem P_0, finding a solution to the problem $P_{0,\lambda}$ is NP-hard and it is thus impossible in practical applications.

Method of frames denoising A straightforward solution is given in the case $p = 2$. The solution of the problem

$$P_{2,\lambda} : \min_{\mathbf{c}} \|\mathbf{x} - \mathbf{\Phi c}\|_2^2 + \lambda E^{(2)}(\mathbf{c}) \tag{3.17}$$

is given by

$$\mathbf{c}^{opt} = \mathbf{\Phi}^T \left(\mathbf{\Phi}\mathbf{\Phi}^T + \lambda \mathbf{I}\right)^{-1} \mathbf{x} \tag{3.18}$$

But similarly to the problem P_2, the solution is not sparse [CDS99].

Basis Pursuit Denoising Chen, Donoho and Saunders, who have considered the exact signal representation case with $p = 1$, called Basis Pursuit, have also considered the signal approximation case, and derived a similar problem for $p = 1$, called Basis Pursuit Denoising [CDS99]

$$P_{1,\lambda} : \min_{\mathbf{c}} \|\mathbf{x} - \mathbf{\Phi c}\|_2^2 + \lambda E^{(1)}(\mathbf{c}) \tag{3.19}$$

It is possible to formulate this problem as a quadratic programming and thus to find a solution using standard quadratic programming techniques, such as Active Set algorithms.

Thresholding algorithms Quadratic programming algorithms that solve Basis Pursuit Denoising require high computational cost. However, when the dictionary has a particular structure, there exist fast algorithms that lead to the solution of $P_{1,\lambda}$. The first example is the case of the orthonormal basis, where the optimal solution of $P_{1,\lambda}$ is obtained by soft-thresholding [Don95]

$$\mathbf{c}^{opt} = \eta_\lambda \left(\mathbf{\Phi}^T \mathbf{x}\right) \tag{3.20}$$

with

$$\eta_\lambda(c) = \begin{cases} c - \frac{\lambda}{2} & , \text{if } c \geq \frac{\lambda}{2} \\ 0 & , \text{if } |c| \leq \frac{\lambda}{2} \\ c + \frac{\lambda}{2} & , \text{if } c \leq \frac{-\lambda}{2} \end{cases} \tag{3.21}$$

Sardy, Bruce and Tseng [SBT00] extend this result to a union of orthonormal basis and proposed an alternate soft-thresholding algorithm, called Block Coordinate Relaxation. It is an iterative algorithm that updates at each iteration the coefficients corresponding to only one orthonormal basis using soft-thresholding. It is shown in [Don95] that this algorithm converges to the optimal solution of $P_{1,\lambda}$. But this algorithm is ineffective when $\lambda \to 0$. More general and similar algorithms are known as iterative thresholding algorithms, they work with any dictionary and converge to a local minimum of $P_{1,\lambda}$, a good review of these algorithms is [EMSZ07].

LASSO A problem similar to Basis Pursuit Denoising ($P_{1,\lambda}$) is considered by Tibshirani et al. and called the Least Absolute Shrinkage and Selection Operator (LASSO) problem [Tib96]. Algorithms that solve this problem include the Least Angle Regression [EJHT04], and more recent and faster algorithms such as e.g. Gradient Projection [FNW07] or Preconditioned Conjugate Gradient [KKL$^+$07].

Regularized FOCUSS Rao, Engan, Cotter, Palmer and Kreutz-Delgado [RRE$^+$03] have considered the more general problem

$$P_{0<p\leq 1,\lambda} : \min_{\mathbf{c}} \|\mathbf{x} - \mathbf{\Phi c}\|_2^2 + \lambda E^{(p)}(\mathbf{c}) \tag{3.22}$$

and proposed several algorithms which are regularized versions of the FOCUSS algorithms.

Bayesian framework Another class of methods that find sparse approximations are based on a Bayesian framework (e.g. [VP07, FTDG08]). The sparseness of the approximation is imposed as a prior on the coefficients, then the approximation is found using Maximum A Posteriori.

Greedy algorithms Finally, another class of methods that find sparse approximations are based on iterative methods that selects one atom at a time, they are known as greedy algorithms. They are very popular due to their performance and their relative low complexity. This is also the class of methods we have chosen for our project, and thus we detail them with more details in the following.

3.3 Greedy algorithms

We have presented in the last section a number of algorithms that find sparse representations/approximations. However, they are only used in very specific applications, and none of these are found in real coding applications. The reason is either they are restricted to a specific dictionary, or they have a limited performance, or they are too complex, or they are too recent.

We present in this section a class of algorithms, called "greedy algorithms" that are able to find sparse approximations with a good compromise in terms of performance and complexity. The greedy algorithms are used in a number of applications including audio coding (e.g. [Vaf04]), image coding (e.g. [FiVVF06]) and video coding (e.g. [NZ97]).

Greedy algorithms are iterative algorithms that solve $P_{0,\lambda}$ at each iteration. However, this local optimization does not lead to a global optimization in the general case. The basic principle of greedy algorithms is as follows. At each iteration, a new atom is added to the approximation, which makes the approximation error tend to zero. The algorithm stops when a user-specified criteria is met. We first describe in the following the simplest greedy algorithm, called Matching Pursuit. Then, we describe variants of this algorithm. We assume in the following that the dictionary is complete.

3.3.1 Matching Pursuit

Matching Pursuit (MP) [MZ93] is the first and simplest greedy algorithm. It selects at each iteration the locally optimal atom i.e. the atom that minimizes the error at the current iteration, and subtract it to the residual.

The algorithm is initialized at iteration $n = 0$ by setting $\mathbf{r}_0 = \mathbf{x}$, where $\mathbf{r}_n = \mathbf{x} - \hat{\mathbf{x}}_n$ is the residual at iteration n and $\hat{\mathbf{x}}_n$ is the approximant at iteration n. We suppose now that we have computed the residual at iteration n, it is then further decomposed by projecting it orthogonally on an atom $\mathbf{g}_{k_n}$

$$\mathbf{r}_n = \langle \mathbf{r}_n, \mathbf{g}_{k_n} \rangle \mathbf{g}_{k_n} + \mathbf{r}_{n+1} \tag{3.23}$$

Since the residual $\mathbf{r}_{n+1}$ is orthogonal to the atom $\mathbf{g}_{k_n}$, we have

$$\|\mathbf{r}_n\|^2 = |\langle \mathbf{r}_n, \mathbf{g}_{k_n} \rangle|^2 + \|\mathbf{r}_{n+1}\|^2 \tag{3.24}$$

At iteration n, the optimal atom index k_n is the one which minimizes the norm of the residual

$$k_n = \underset{1 \leq k \leq K}{\operatorname{argmin}} \|\mathbf{r}_{n+1}\|^2 \tag{3.25}$$

The minimum is unique and the optimal atom index is given by

$$k_n = \underset{1 \leq k \leq K}{\operatorname{argmax}} |\langle \mathbf{r}_n, \mathbf{g}_k \rangle| \tag{3.26}$$

which corresponds to the atom that is most correlated with r_n and the corresponding coefficient is

$$c_{k_n} = \langle \mathbf{r}_n, \mathbf{g}_{k_n} \rangle \tag{3.27}$$

and the approximant at iteration n is given by

$$\hat{\mathbf{x}}_n = \sum_{i=0}^{n-1} c_{k_i} \mathbf{g}_{k_i} \tag{3.28}$$

The algorithm is stopped when a condition on the number of atoms or on the residual is met. A simple pseudo-code of the MP algorithm is given in **Algorithm 4**.

Algorithm 4: Matching Pursuit

Input: The signal $\mathbf{x}$ and the dictionary Φ

Output: The vector of coefficients $\mathbf{c}$ and the residual $\mathbf{r}$ such that $\mathbf{x} = \Phi\mathbf{c} + \mathbf{r}$

1 Initialization: $\mathbf{r} = \mathbf{x}$, $\mathbf{c} = \mathbf{0}$;
2 **repeat**
3 Inner products: $\mathbf{a} = \Phi^T \mathbf{r}$;
4 Find maximum: $k_{max} = \underset{1 \leq k \leq K}{\operatorname{argmax}} |a_k|$;
5 Update coefficients: $c_{k_{max}} = c_{k_{max}} + a_{k_{max}}$;
6 Update residual: $\mathbf{r} = \mathbf{r} - a_{k_{max}} \Phi_{k_{max}}$;
7 **until** *a stopping condition is met* ;

We now recall the convergence of MP [MZ93]. We first define the correlation ratio of a vector $\mathbf{f}$ with respect to the dictionary $\mathcal{D}$

$$\lambda(\mathbf{f}) = \max_{1 \leq k \leq K} \frac{|\langle \mathbf{f}, \mathbf{g}_k \rangle|}{\|\mathbf{f}\|} \tag{3.29}$$

At iteration n, MP chooses the atom k_n and we have

$$\|\mathbf{r}_{n+1}\|^2 = \|\mathbf{r}_n\|^2 - |\langle \mathbf{r}_n, \mathbf{g}_{k_n} \rangle|^2 = \|\mathbf{r}_n\|^2 \left(1 - \lambda^2 (\mathbf{r}_n)\right) \tag{3.30}$$

It is proved in [MZ93] that $\lambda(\mathbf{f})$ is strictly larger than a positive constant

$$\exists \beta > 0 \text{ such that } \lambda(\mathbf{f}) > \beta \ , \forall \mathbf{f} \in \mathcal{R}^N \tag{3.31}$$

β is a constant that depends only on the dictionary, it can be interpreted as the cosine of the maximum angle between any direction in $\mathcal{R}^N$ and the closest element of the dictionary [MZ93]. We then have

$$\|\mathbf{r}_{n+1}\|^2 \leq \|\mathbf{r}_n\|^2 \left(1 - \beta^2\right) \tag{3.32}$$

and

$$\|\mathbf{r}_n\| \leq \|\mathbf{x}\| \left(1 - \beta^2\right)^{n/2} \tag{3.33}$$

This guarantees that MP converges to zero, and it even says that it converges exponentially. However, this upper bound on the residual energy is pessimistic as it generally decreases much faster in the first iterations.

3.3.2 Variants of Matching Pursuit

Orthogonal Matching Pursuit Matching Pursuit selects at each iteration the optimal atom, which is only locally optimal. The first selected atoms are not guaranteed to be optimal globally. Orthogonal Matching Pursuit (OMP) [PRK93] is a variant of MP that at each iteration, optimizes the weights of the already selected atoms by projecting orthogonally the signal onto these atoms. At iteration n, OMP selects an atom using the same criterion as MP, and then solves the following least-squares problem

$$\min_{\mathbf{c}_{\Gamma_n}} \|\mathbf{x} - \mathbf{\Phi}_{\Gamma_n} \mathbf{c}_{\Gamma_n}\|_2 \tag{3.34}$$

where Γ_n is the set of indices of the selected atoms up to and including iteration n

$$\Gamma_n = \{k_i, 1 \leq i \leq n\} \tag{3.35}$$

This problem has a unique solution which is

$$\mathbf{c}_{\Gamma_n} = \mathbf{\Phi}^+_{\Gamma_n} \mathbf{x} \tag{3.36}$$

Algorithm 5: Orthogonal Matching Pursuit

Input: The signal $\mathbf{x}$ and the dictionary Φ
Output: The vector of coefficients $\mathbf{c}$ and the residual $\mathbf{r}$ such that $\mathbf{x} = \Phi\mathbf{c} + \mathbf{r}$

Initialization: $\mathbf{r} = \mathbf{x}$, $\mathbf{c} = \mathbf{0}$, $\Gamma = \emptyset$;
repeat
  Inner products: $\mathbf{a} = \Phi^T \mathbf{r}$;
  Find maximum: $k_{max} = \underset{1 \leq k \leq K}{\operatorname{argmax}} |a_k|$;
  Update indices set: $\Gamma = \Gamma \cup k_{max}$;
  Update coefficients: $\mathbf{c}_\Gamma = \mathbf{\Phi}^+_\Gamma \mathbf{x}$;
  Update residual: $\mathbf{r} = \mathbf{x} - \Phi\mathbf{c}$;
until *a stopping condition is met* ;

where $\mathbf{\Phi}_{\Gamma_n}^+$ is the pseudo-inverse of $\mathbf{\Phi}_{\Gamma_n}$. A simple pseudo-code of the OMP algorithm is given **Algorithm 5**.

As compared to MP, OMP needs one additional operation, the computation of the pseudo-inverse of $\mathbf{\Phi}_\Gamma$, which is very costly if computed directly. Efficient implementations of this operation exist, such as [MDZ94] which is based on QR factorization, and [CARKD99] which is based on Cholesky factorization.

Gradient Pursuits Blumensath and Davies proposed in [BD08] two variants of Matching Pursuit based on directional optimization: Gradient Pursuit (GP) and Approximate Conjugate Gradient Pursuit (ACGP). At iteration n, they select the best atom as in MP. Then, instead of updating $c_{k_{max}}$ by adding $a_{k_{max}}$ (see Alg. 6), they propose a directional update, which is different for GP and ACGP. These algorithms outperform MP with performances close to OMP. The advantage compared to OMP is less computational complexity and memory demands, which make them usable in practical applications. The pseudo-code of the gradient pursuits is given below

Algorithm 6: Gradient Pursuits

Input: The signal $\mathbf{x}$ and the dictionary Φ

Output: The vector of coefficients $\mathbf{c}$ and the residual $\mathbf{r}$ such that $\mathbf{x} = \Phi\mathbf{c} + \mathbf{r}$

Initialization: $\mathbf{r} = \mathbf{x}$, $\mathbf{c} = \mathbf{0}$, $\Gamma = \emptyset$;
repeat
  Inner products: $\mathbf{a} = \Phi^T\mathbf{r}$;
  Find maximum: $k_{max} = \underset{1 \le k \le K}{\operatorname{argmax}} |a_k|$;
  Update indices set: $\Gamma = \Gamma \cup k_{max}$;
  Calculate update direction $\mathbf{d}_\Gamma$;
  Calculate corresponding vector: $\mathbf{h} = \mathbf{\Phi}_\Gamma \mathbf{d}_\Gamma$;
  Calculate corresponding coefficient: $b = \langle \mathbf{r}, \mathbf{h} \rangle / \|\mathbf{h}\|_2^2$;
  Update coefficients: $\mathbf{c}_\Gamma = \mathbf{c}_\Gamma + b\mathbf{d}_\Gamma$;
  Update residual: $\mathbf{r} = \mathbf{r} - b\mathbf{h}$;
until *a stopping condition is met* ;

Other variants Other variants of MP include for example Weak (Orthogonal) Matching Pursuit [Tem00], that relaxes the atom selection rule in (O)MP and tree-based Pursuit [JVF06], that uses a tree structure to reduce complexity in MP.

3.4 Dictionaries used in coding applications

One of the main issues in sparse representations/approximations is the design of the dictionary. A first possibility is to learn the dictionary from a large collection of signals (e.g. [EA06, MJV+07]). Another possibility is to use deterministic dictionaries. An exhaustive review of all deterministic dictionaries is out of the scope of this chapter. Instead, as we are interested in coding applications, we restrict our review to the dictionaries used in audio/image/video coding applications. We first review some of well-known orthogonal dictionaries, which are used widely in standard transform coding. We then review some overcomplete dictionaries that have been successfully applied to coding applications.

3.4.1 Orthogonal dictionaries

The most basic orthogonal dictionary is the Dirac dictionary. This corresponds to a Pulse Code Modulation (PCM) signal representation. For a signal of length N, the atoms are given by

$$g_m(n) = \begin{cases} 1 & \text{if } m = n \\ 0 & \text{otherwise} \end{cases}, \; 0 \leq n, m < N \tag{3.37}$$

When dealing with audio signals, we obtain much sparser representations using time-frequency dictionaries based on local cosine functions. The reason is that audio signals are essentially composed by sinusoidal components. The most widely used orthogonal time-frequency dictionary in audio coding is the MDCT introduced in Sec. 2.2.1. The atoms corresponding to a MDCT with a fixed window length $2L$ are given by

$$g_{p,k}(n) = w_p(u)\sqrt{\frac{2}{L}}\cos\left[\frac{\pi}{L}\left(u + \frac{L}{2} + \frac{1}{2}\right)\left(k + \frac{1}{2}\right)\right] \tag{3.38}$$

and

$$u = n - \left(p - \frac{1}{2}\right)L \tag{3.39}$$

and $w_p(u)$ the analysis window defined on $0 \leq u < 2L$. Refer to Sec. 2.2.1 for details. Examples of MDCT atoms for a fixed window size MDCT are on Fig. 3.1.

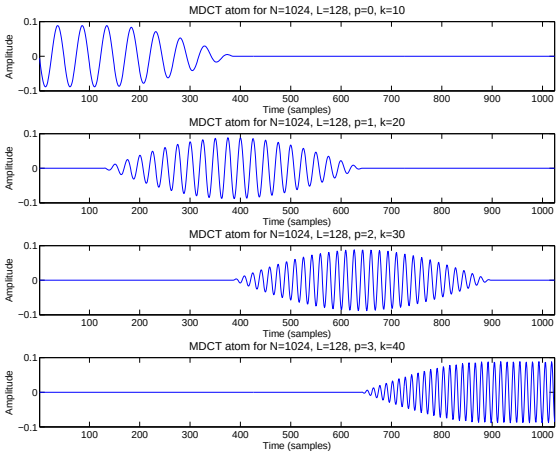

Figure 3.1: *Example of MDCT atoms for a signal of length* $N = 1024$ *and a MDCT of fixed window size* $2L = 256$ *and a sine-based window shape.*

When dealing with images, either frequency dictionaries such as the Discrete Cosine Transform (DCT) or time-scale dictionaries such as the Discrete Wavelet Transform (DWT) are preferred. As an example, the JPEG image coder uses a DCT on small blocks of 8x8 or 16x16 pixels. The atoms corresponding to a 2-dimension DCT with size $M \times N$ are defined as:

$$g_{p,q}(m,n) = \alpha_p \beta_q \cos\left[\frac{\pi}{M}\left(m + \frac{1}{2}\right)\right]\cos\left[\frac{\pi}{N}\left(n + \frac{1}{2}\right)\right] \tag{3.40}$$

with

$$\alpha_p = \begin{cases} \sqrt{1/M} & , \; p = 0 \\ \sqrt{2/M} & , \; p > 0 \end{cases} \tag{3.41}$$

and

$$\beta_q = \begin{cases} \sqrt{1/N} & , q = 0 \\ \sqrt{2/N} & , q > 0 \end{cases} \tag{3.42}$$

Fig. 3.2 shows all atoms corresponding to a DCT of size 8×8 (as in JPEG).

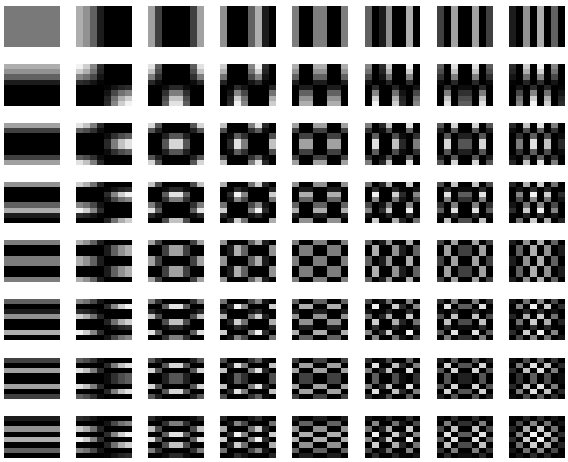

Figure 3.2: *All* 64 *atoms corresponding to a DCT of size* 8×8

DWT is used in more recent standard image coding. A well-known example is JPEG2000 [ISO04a] that uses biorthogonal wavelets, either Daubechies 5/3 wavelets or LeGall 5/3 wavelets. A good reference for wavelet-based transforms is [Mal98].

3.4.2 Overcomplete dictionaries

The earliest and now widely used overcomplete time-frequency dictionary is the Gabor dictionary [Gab47], whose atoms are defined as complex exponential modulated Gaussian windows w and given for a 1-D continuous signal

$$g_{s,u,f}(t) = w\left(\frac{t-u}{s}\right) e^{2j\pi ft} \tag{3.43}$$

where u is the time shift, s is the scale and f is the frequency. In practice, discretized Gabor atoms are used. Gabor atoms were originally defined using a Gaussian window, but they now refer more generally to the modulated version of any window. It is important to remark that Gabor atoms are complex atoms, and consequently a real signal is usually represented by pair of conjugate atoms [MZ93, Gri99, Goo01]. 1-D Gabor atoms are used in parametric audio coding (e.g. [LS98, VM00, PM00, dSO02, Vaf04]), to model the sinusoidal components of an audio signal. 2-D Gabor atoms are also used in video coding (e.g. [NZ97, ASMN$^+$99]), to model the motion residual signals. Figure 3.4.2 shows examples of 1D and 2D gabor atoms (these are discrete signals).

Gabor atoms are very similar to MDCT/DCT atoms as they are all based on local cosine functions. However, Gabor atoms are more general and allow more freedom in the parameters of the local cosine functions as compared to the MDCT/DCT: a free phase, a better frequency resolution, and a freedom in the window size and shape. Such a freedom allows to design large dictionaries where there is more chance to find atoms that best match the signal to model, and consequently Gabor dictionary generally produces sparser approximations than MDCT/DCT.

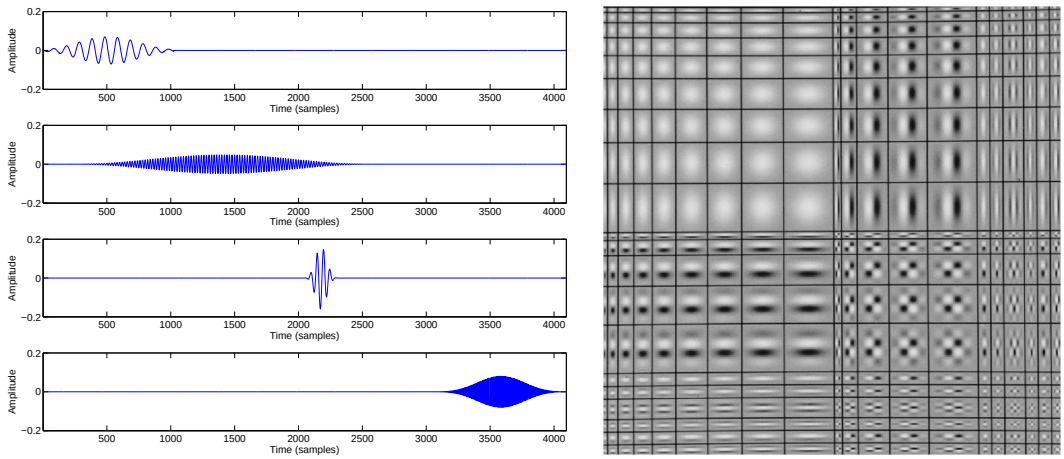

Figure 3.3: Examples of 1D and 2D Gabor atoms

Other time-frequency atoms well-suited for audio signals are the so-called harmonic atoms, which are composed by a linear combination of several Gabor atoms whose frequencies are in harmonic relation. They defined, for a 1-D continuous signal, as

$$h_{s,u,f_0,\mathbf{a},\phi} = \sum_k a_k e^{j\phi_k} g_{s,u,k.f_0} \tag{3.44}$$

where $g_{s,u,k.f_0}$ are Gabor atoms, a_k the partial amplitudes, and ϕ_k the partial phases. Harmonic atoms are used for example in the standard MPEG-4 HILN [PM00], and also in more recent and experimental very low bitrate parametric audio coding [VP07, CRLD07].

For modeling images, another well suited dictionary is generated by applying meaningful geometric transformations to one or more generating functions. The transformations are of 3 types: translation, rotation and anisotropic scaling. The generated atoms efficiently model the edges of the images. Such a dictionary is successfully applied to low bitrate image coding in [FiVVF06] and to hybrid video coding in [GMPV04].

3.5 Conclusions

We have introduced in this chapter sparse signal representations. The basic principle of sparse signal representations is to model a signal as a sum of small number of elementary functions (called atoms) which are chosen among a collection (called dictionary) of arbitrary size. We have then defined the basic problems of such representations. One the one hand, we have seen that the problems have a straightforward solution if the dictionary is an orthogonal basis (this is the case for example of an MDCT in transform coding). On the other hand, sparser solutions are generally obtained when using overcomplete dictionaries (for example Gabor atoms used in sinusoidal modeling) because there are more chances of finding atoms that best match the signal to model in a larger dictionary, but in this case the problems are also much more complicated. We have then reviewed some algorithms that find sparse representations/approximations in overcomplete dictionaries. The main issue of such algorithms is the computational complexity and we have then focused on a class of algorithms that allow a good trade-off between performance and complexity, called greedy algorithms and including the well-known Matching Pursuit.

Chapter 4

Signal representation in a union of MDCT bases

Abstract

This chapter proposes a new signal representation approach for audio coding. It is a generalization of the standard MDCT where several MDCT bases with different scales are used simultaneously. A signal is approximated over a union of MDCT bases using the matching pursuit algorithm. We show that this new approach gives sparser approximations than a single MDCT but at a higher computational cost. We also proposes a new modified matching pursuit algorithm that allows the reducing of the pre-echo artifact. In this chapter, we restrain our study to the signal representation only. We will see how it is applied to audio coding in the next chapters.

Contents

4.1 Introduction

We have presented in the first chapter a review of existing signal represention methods used in modern speech and audio coders. We have then presented in the second chapter a particular class of signal representations, named sparse signal representations, used in numerous applications and particularly in many coding applications such as audio, image or video coding.

We propose in this chapter a new signal representation for audio coding that uses the methods of the sparse signal representations. This new signal representation is based on a union of a number of MDCT bases with different scales. Our approach is related to and different from existing methods in several ways. First, it is related to MDCT-based coding (see section 2.2.1, e.g. MPEG-4 AAC) and could be seen as a generalization of the transform approach since it is based on a simultaneous use of a union of MDCT bases. Second, our approach is also related to sinusoidal modeling based coding (see section 2.2.2, e.g. MPEG-4 SSC). We model the signal as a sum of multi-scale sinusoidal atoms which is closely related to multiresolution sinusoidal modeling. In sinusoidal modeling based audio coding, the sinusoidal model tries to match the sinusoidal content of the signal as closely as possible, which is done using complex sinusoidal atoms with a precise estimate of amplitude, frequency and phase and a post-processing stage that builds sinusoidal tracks. In our approach, we extract real sinusoidal atoms with only an amplitude parameter, and so we do not have to transmit any phase parameter. Contrary to the parametric approach, our frequencies are sampled from a limited range (the FFT size equals the analysis window length), which permits transmission of frequency without requantizing. Moreover, since the sinusoidal decompositions used in parametric audio coding extract a limited number of sinusoids from the signal and model the residual as noise (plus perhaps transients modeling). Clearly, the sinusoidal decompositions only model a subspace of the signal which limits their performance at high bitrates. Our approach is more general as it models the signal entirely with sinusoidal atoms, which is feasible here at reasonable coding cost because the set of time-frequency atoms has a limited size and consequently the cost to encode the index of the selected atoms is not prohibitive. As a consequence, it has the possibility of providing transparency at high bitrates.

In this chapter, we restrain our study to the signal representation only. We will see how it is applied to audio coding in the next chapters. We first formalize the signal model (the dictionary) in Section 4.2. Then, we describe the algorithm that finds sparse approximations and present a few results in Sections 4.3 and 4.4. And finally, we raise a problem introduced by the standard decomposition algorithm and propose a solution to solve it in Section 4.5.

4.2 Signal model

We propose a new signal representation method where a signal is approximated by a linear combination of atoms. These atoms are chosen among a dictionary composed by a union of MDCT bases. In this section, we first explain why we have chosen such a dictionary. Then, we formalize the signal model.

4.2.1 Motivations

State-of-the-art audio coders that provide (near-)transparent quality are based on MDCT (ex. AAC). We have seen in Sec. 2.2.1 that MDCT with a window length L is an orthogonal transform, and thus provides perfect reconstruction with critical sampling. However, as explained in Sec. 2.2.1, using a fixed window length provides a time-frequency transform with a fixed time-frequency resolution. Either the window size is long (e.g. 2048 samples) and the transform has a

good frequency resolution but a poor time resolution, or the window size is short (e.g. 256 samples) and the transform has a good time resolution but a poor frequency resolution. The solution used in state-of-the-art audio coders and presented in Sec. 2.2.1 is to use an adaptive window size. Usually two window sizes are used, one long for stationary parts and one short for transients; the decision to use one or the other is based on an energy or a perceptual entropy criterion. This time-varying MDCT is still an orthogonal transform if appropriate window shapes are used, particularly for the transition between long and short window sizes. A more general approach has been proposed by Niamut et al. [Nia06], where a MDCT with several window sizes is used (e.g. 256, 512, 1024, 2048). They propose a RD-optimized algorithm that partitions the signal in segments, where in each segment a fixed window size is used. With this approach, the time-varying MDCT is still an orthogonal transform, under the conditions that appropriate window shapes are chosen.

The existing MDCT-based approaches use either a fixed resolution or an adaptive resolution. However, even for the case of an adaptive resolution, the resolution is still fixed inside a given segment. This is not optimal for sound signals containing simultaneous components localized both in time and frequency. For instance, drums on top of long sustained notes would force a time segmentation algorithm to break up the long notes into smaller pieces. It would be better to be allowed to superimpose components with different time-frequency resolution. We thus propose in the following the simultaneous use of several MDCT with different window sizes. Each MDCT has a fixed window size, but the simultaneous use of several MDCT allows the modeling of simultaneous components which can have different time-frequency tradeoffs.

To better illustrate this time-frequency tradeoff, we consider an extract of a glockenspiel signal. This signal is characterized by its strong attacks which are localized in time, and its long stationary components which are localized in frequency. Fig. 4.1 shows the MDCT pseudo-spectrogram of the glockenspiel extract signal for 8 window sizes: 128, 256, 512, 1024, 2048, 4096, 8192 and 16384. It is obvious to see that a MDCT with a long window size has not enough time resolution to model efficiently the sharp attacks of the glockenspiel signal. On the other hand, a small window size MDCT has not enough frequency resolution to model efficiently the long stationary parts. This second issue is better shown in Fig. 4.2. In this figure, only the spectrum of a single frame of each MDCT is plotted. The stationary parts of the glockenspiel correspond here to the peaks of the spectrum which are sharper for a longer window.

4.2.2 Model formalization

Let us now formalize the signal model. We use a union of M MDCT bases with fixed window size as defined in Section 2.2.1. The analysis window lengths are defined as increasing power of two

$$L_m = L_0 2^m. \tag{4.1}$$

Considering a signal $\mathbf{x}$ with length N samples, it is approximated as

$$\mathbf{x} = \mathbf{\Phi c} + \mathbf{r} \tag{4.2}$$

with $\mathbf{r}$ the residual, $\mathbf{c}$ the vector of coefficients of length MN and $\mathbf{\Phi}$ the synthesis matrix of size $N \times MN$ defined as the concatenation of M matrices T_m of size $N \times N$, each matrix T_m corresponding to one MDCT, as defined in equations (2.14), (2.15) and (2.16). The synthesis matrix $\mathbf{\Phi}$ is defined explicitly as:

$$\mathbf{\Phi}\left(n, mN + pK_m + k\right) = g_{m,p,k}(n) \ , \ 0 \leq m < M \ , \ 0 \leq p < P_m \ , \ 0 \leq k < K_m \ , \ 0 \leq n < N \tag{4.3}$$

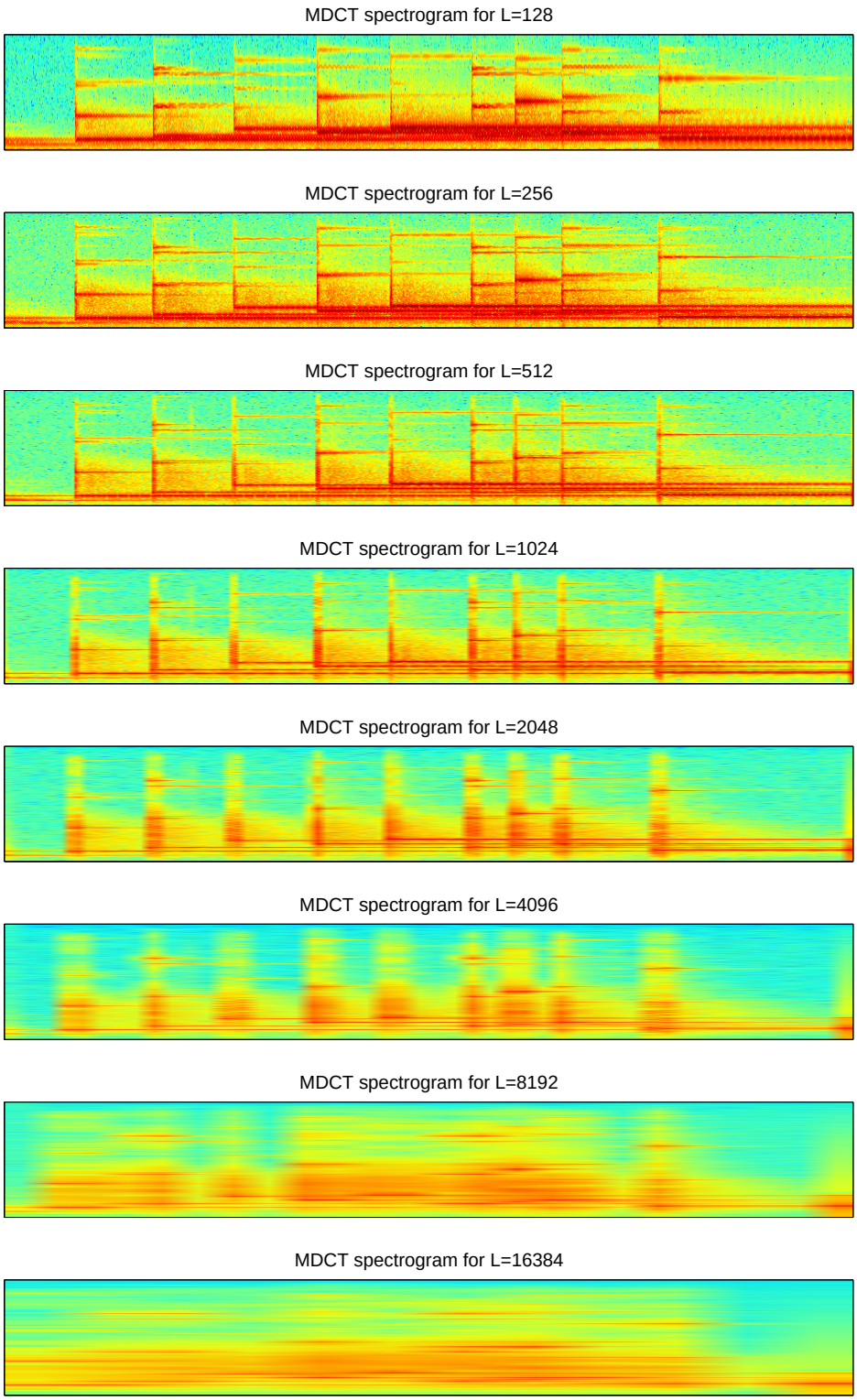

Figure 4.1: *MDCT spectrogram of an extract of the glockenspiel signal for several window sizes.*

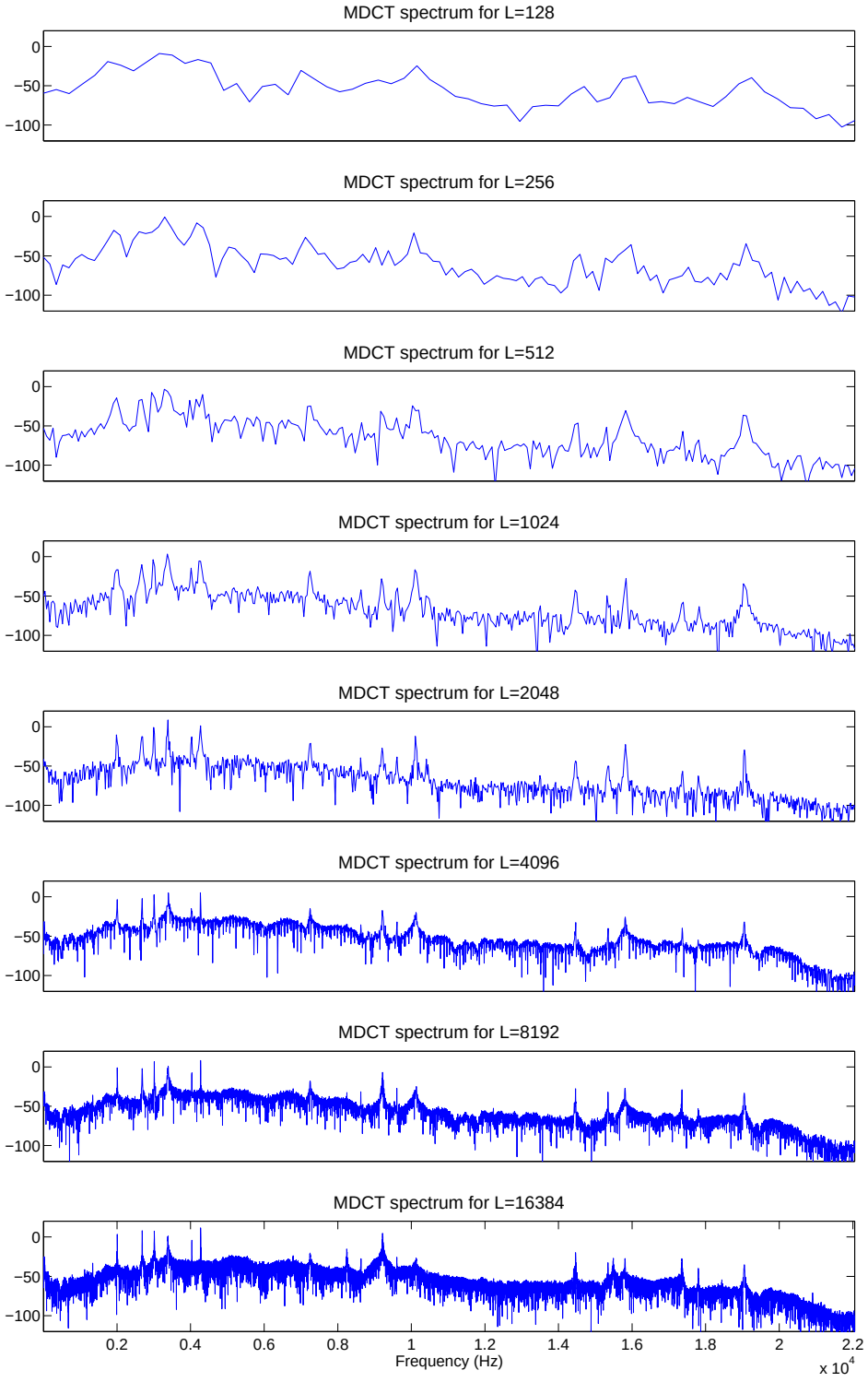

Figure 4.2: *MDCT spectrum of one frame of the glockenspiel signal for several window sizes.*

where P_m is such that $N = P_m K_m$ and is the number of segments of length $K_m = L_m/2$, and

$$g_{m,p,k}(n) = w_{m,p}(u)\sqrt{\frac{2}{K_m}}\cos\left[\frac{\pi}{K_m}\left(u + \frac{K_m}{2} + \frac{1}{2}\right)\left(k + \frac{1}{2}\right)\right] \tag{4.4}$$

and

$$u = n - \left(p - \frac{1}{2}\right)K_m \tag{4.5}$$

The window $w_{m,p}(u)$ is defined over $0 \leq u < L_m$ and verifies Eq. 2.22, 2.23 and 2.24 (see Sec. 2.2.1). Figure 4.3 shows the sine-based windows $w_{m,p}(u)$ for $M = 4$ MDCT bases.

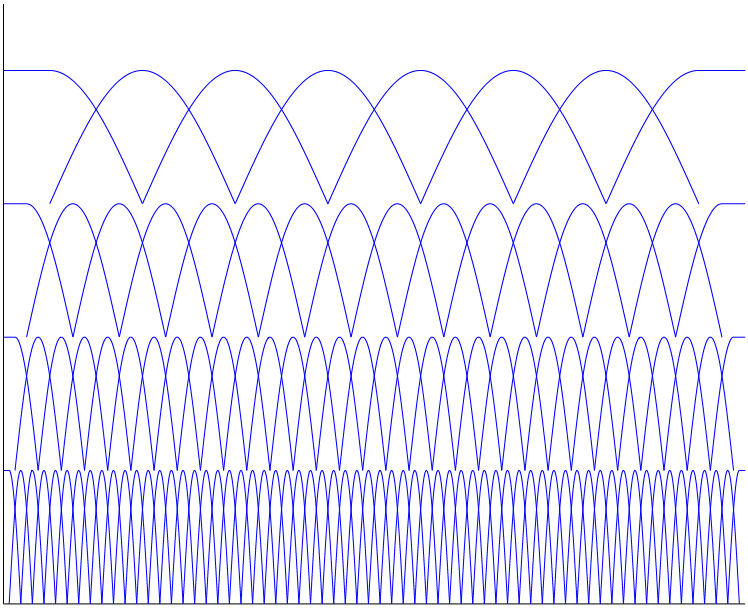

Figure 4.3: *Analysis windows for 4 MDCT bases*

4.3 Fast implementation of Matching Pursuit in a union of MDCT bases

We have formalized in the last section the dictionary that we use to approximate a signal. Now, given a signal $\mathbf{x}$ and the dictionary $\mathbf{\Phi}$, the problem is now to find a sparse approximation $\mathbf{c}$ such that we have $\mathbf{x} = \mathbf{\Phi c} + \mathbf{r}$ where $\mathbf{r}$ is the residual. We have seen in section 3.2 that if the dictionary is overcomplete then there is an infinity of possible approximations, and that finding the approximation with the minimum number of non-zero terms (minimum 0-like-norm) is a NP-hard problem. However, we have also seen that there exists a number of algorithms that find "good" sparse approximations. Particularly, we have presented in section 3.3 a class of algorithms that give a good compromise between complexity and performance: these are called the greedy algorithms. The simplest Greedy Algorithm is Matching Pursuit and there exists a fast implementation of Matching Pursuit for time-frequency dictionaries called Matching Pursuit ToolKit [Krs06] (MPTK) [1]. The availability of a fast implementation is one of the main reason we decided to choose Matching

[1] MPTK is released under the GNU/GPL license and is distributed at the following address: http://gforge.inria.fr/projects/mptk

Pursuit and slightly modified versions of Matching Pursuit in this thesis.

In this section, we first recall the Matching Pursuit algorithm and the main implementation issues, then we introduce the Matching Pursuit ToolKit, and finally we present a fast implementation of the MDCT we have implemented within the MPTK framework.

4.3.1 MP: Naive implementation

We first recall the matching pursuit algorithm. The pseudo-code of the algorithm is given below. There are two critical operations that would require high ressources if implemented naively.

Algorithm 7: Matching Pursuit

Input: The signal $\mathbf{x}$ and the dictionary Φ
Output: The vector of coefficients $\mathbf{c}$ and the residual $\mathbf{r}$ such that $\mathbf{x} = \Phi\mathbf{c} + \mathbf{r}$

1 Initialization: $\mathbf{r} = \mathbf{x}$, $\mathbf{c} = \mathbf{0}$;
2 **repeat**
3 Inner products: $\mathbf{a} = \Phi^T\mathbf{r}$;
4 Find maximum: $k_{max} = \underset{1 \leq k \leq K}{\operatorname{argmax}} |a_k|$;
5 Update coefficients: $c_{k_{max}} = c_{k_{max}} + a_{k_{max}}$;
6 Update residual: $\mathbf{r} = \mathbf{r} - a_{k_{max}}\Phi_{k_{max}}$;
7 **until** *a stopping condition is met* ;

These two operations correspond to the lines 3 and 4 of the pseudo-code, and they are detailed in the following

1. The inner products calculation: a naive implementation would require the storage of MN^2 values (the dictionary matrix Φ), and the computation of MN^2 multiplications and $MN(N-1)$ additions per iteration. As an example, we consider a signal of length $N = 88200$ samples (2 seconds) and a $M = 8$ times overcomplete dictionary. A naive implementation would thus require the storage of $6,2 \cdot 10^{10}$ real values (more than 400 Gigs of memory storage) and the computation of $6,2 \cdot 10^{10}$ multiplications and $6,2 \cdot 10^{10}$ additions.

2. The search of the maximum: a naive implementation would require a memory scan of MN values per iteration. For the example given above, this corresponds to a memory scan of 705600 values.

A naive implementation would thus require very high ressources which is not desirable for coding applications. We then discuss in the following a fast implementation of Matching Pursuit.

4.3.2 MP: Fast implementation

We introduce here MPTK, a fast implementation of MP written in C++ and released under the GNU/GPL license. MPTK is designed for time-frequency dictionaries and is mainly based on three tricks that allow a huge gain in terms of computational complexity and memory requirements as compared to a naive implementation. These three tricks are detailed in the following.

1. The first trick is to use a fast transform for the computation of the inner products between the residual and the time-frequency atoms. This avoids the storage of the dictionary matrix

Φ (MN^2 values) and greatly reduces the number of multiplications and additions needed for the inner products calculation. In the case of a union of MDCT bases, one MDCT is called a "block" in MPTK, and for each block, the set of atoms with same support (same m and p) is called a "frame". In MPTK, the inner products are then computed separately for each frame of each block using a fast transform. The fast implementation of the MDCT we have implemented in MPTK is described later in 4.3.3.

2. The second trick uses the fact that time-frequency atoms have a finite support, whose length is small compared to the signal dimension. At each iteration, MP selects one atom of finite length support. Consequently, the residual is modified only on a finite support (line 6 of the pseudo-code), the rest of the residual does not change. And only a subset of the inner products between the atoms and the residual change from one iteration to the next. MPTK thus computes at each iteration only the inner products of the frames whose support overlap the support of the atom selected at the previous iteration. As an example, considering a signal of dimension N, and a union of 4 MDCT bases with window sizes 256,512,1024,2048. We suppose that at iteration n, MP selects one atom of length 1024. At the next iteration, only the inner product of a small number of atoms needs updating: 1×1024 atoms of length 2048, 3×512 atoms of length 1024, 6×256 atoms of length 512 and 10×128 atoms of length 256. This makes a total of 5376 atoms, instead of $4N$ atoms. If N is large, this makes a huge difference.

3. The third trick also uses the fact that time-frequency atoms have a finite support to reduce the computation time needed for searching the maximum (line 4 of the pseudo-code). Instead of storing all the inner product absolute values and searching for the maximum among these values, only the maximums of the absolute value of the inner products of each frame of each block are stored. A tree is then built on top of these values in such a way that the top of tree corresponds to the searched maximum. Consequently, only a few of the tree values are scanned and modified at each iteration. Refer to [Krs06] for a more detailed explanation. This trick significantly reduces the computation time needed by the maximum search.

It should be noted that the only drawback of using MPTK is that it cannot deal with edge atoms. But this problem is easily avoided in the rest of this thesis by zero-padding the signal at both edges. The fast Matching Pursuit algorithm as implemented in MPTK is described below:

4.3.3 Fast Implementation of the MDCT

We present here a fast MDCT implementation we have included in MPTK. The fast implementation of the MDCT avoids the storage of the matrix Φ and reduces the complexity of the inner products calculation. This implementation is detailed in the following. Considering a N-length signal $\mathbf{x}$, and a frame p of block m. The inner products between the signal $\mathbf{x}$ and the atoms $\mathbf{g}_{m,p,k}, k = 0, ..., K_m$ are given by

$$\mathrm{ip}_{p,m}(k) = \sum_{n=0}^{N-1} x(n) g_{m,p,k}(n) \tag{4.6}$$

$$= \sum_{u=0}^{L_m-1} \tilde{x}(u) w_{m,p}(u) \sqrt{\frac{2}{K_m}} \cos\left[\frac{\pi}{K_m}\left(u + \frac{K_m}{2} + \frac{1}{2}\right)\left(k + \frac{1}{2}\right)\right] \tag{4.7}$$

Algorithm 8: Fast Matching Pursuit

Input: The signal $\mathbf{x}$ and the dictionary Φ
Output: The vector of coefficients $\mathbf{c}$ and the residual $\mathbf{r}$ such that $\mathbf{x} = \Phi\mathbf{c} + \mathbf{r}$

```
Initialization: r = x, c = 0;
Initialize support S;
repeat
    for Each block in the dictionary do
        Find the set of frames P whose support overlap with support S;
        for For each frame in P do
            Compute the inner products using a fast transform;
            Compute the absolute value;
            Store the maximum in the tree;
        end
    end
    Update the tree;
    Find maximum: k_max = top of the tree;
    Compute inner products of the corresponding frame (coefficient a_{k_max});
    Build atom waveform Φ_{k_max};
    Update coefficients: c_{k_max} = c_{k_max} + a_{k_max};
    Update residual: r = r − a_{k_max} Φ_{k_max};
    Update support: S = Support(Φ_{k_max});
until a stopping condition is met ;
```

with $\tilde{x}(u) = x(u + \left(p - \frac{1}{2}\right) K_m), u = 0, ..., L_m - 1$. Now, if we write the cosine function as the real part of a complex exponential, we have

$$\mathrm{ip}_{p,m}(k) = \Re\left(\mathrm{ipc}_{p,m}(k)\right) \tag{4.8}$$

with

$$\mathrm{ipc}_{p,m}(k) = \sum_{u=0}^{L_m-1} \tilde{x}(u) w_{m,p}(u) \sqrt{\frac{2}{K_m}} \exp\left[-j\frac{\pi}{K_m}\left(u + \frac{K_m}{2} + \frac{1}{2}\right)\left(k + \frac{1}{2}\right)\right] \tag{4.9}$$

that can be simplified as

$$\mathrm{ipc}_{p,m}(k) = \alpha_m(k) \sum_{u=0}^{L_m-1} \beta_{m,p}(u) w_{m,p}(u) \tilde{x}(u) \exp\left[-j\frac{2\pi uk}{L_m}\right] \tag{4.10}$$

with

$$\alpha_m(k) = \sqrt{\frac{2}{K_m}} \exp\left[\left(\frac{K_m}{2} + \frac{1}{2}\right)\left(k + \frac{1}{2}\right)\right] \tag{4.11}$$

$$\beta_{m,p}(u) = \exp\left[-j\frac{\pi}{L_m}\right] \tag{4.12}$$

We recognize in equation (4.10) the expression of a DFT which is efficiently implemented using a FFT. We summarize the fast MDCT below:

1. Window the input signal $\mathbf{x}$ using the window $w_{m,p}$.

2. Multiply the windowed signal samples by the pre-twiddle coefficients $\beta_{m,p}(u)$.
3. Compute the FFT of the resulting signal of length L_m.
4. Multiply the FFT output by the post-twiddle coefficients $\alpha_m(k)$.
5. Take the real part.

It should be noted that there exists more efficient implementations of the MDCT (e.g. [DMP91]) but the implementation we have chosen has the advantage to provide a unique algorithm for the MDCT (algorithm above), the MDST (imaginary part instead of real part at step 5), and the MCLT (complex output at step 4).

Fig. 4.4 shows the computation time in function of the target SNR needed to approximate a simple signal composed by 512 zeros + 512 samples of white noise + 512 zeros. The dictionary is a union of 4 MDCT bases with window sizes 32, 64, 128 and 256. The naive implementation of MP has been implemented using the Matlab language. These results show that the fast implementation is more than 200 times faster than the naive implementation.

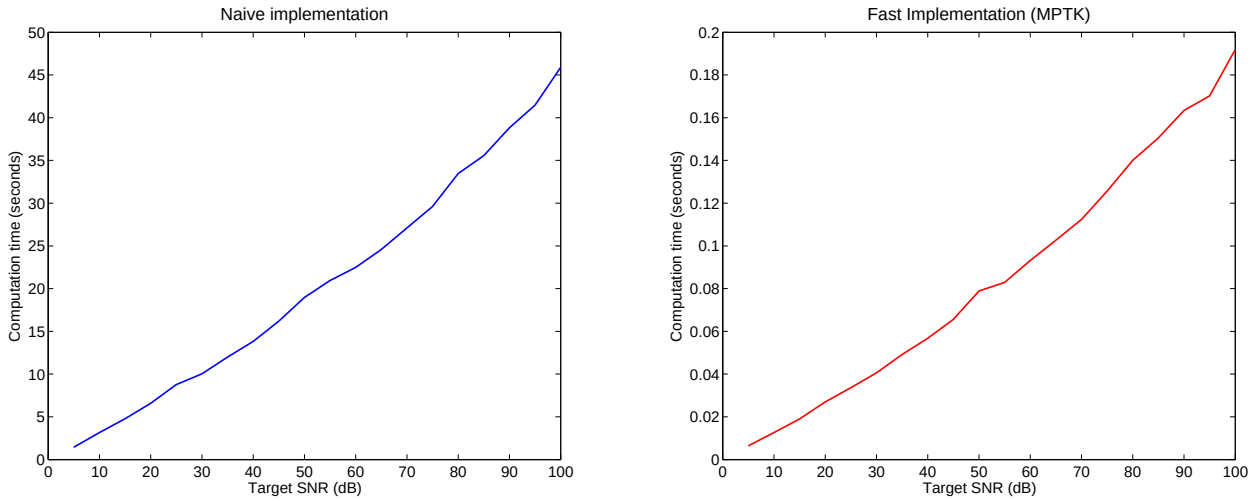

Figure 4.4: *Computation times.*

4.4 Experiment

We have formalized in the first section the dictionary that we use to approximate a signal: the union of several MDCT bases. We have then described in the next section a fast implementation of the decomposition algorithm we have chosen: the Matching Pursuit ToolKit (MPTK). We will now present in this section the results of a few experiments using Matching Pursuit in a union of MDCT bases.

4.4.1 Experimental setup

Test sounds We use the signals described in the appendix A. These are five signals, from the dataset of [ISO03]: bagp, gloc, harp, orch, popm. Each signal is approximately 10 seconds long. We resampled the signals from 48 khz to 44.1 khz and kept only the left channel.

Dictionary We compare in this section several dictionaries: a single MDCT basis with window size 2048 (1XMDCT); a union of 4 MDCT bases with window sizes from 512 to 4096 (4xMDCT); a union of 8 MDCT bases with window sizes from 128 to 16384 (8xMDCT) and a union of 12 MDCT bases with window sizes from 64 to 65536 (12xMDCT).

Algorithm We use the MP algorithm as implemented in MPTK and described in the previous section. The signals are zero-padded with 65536 null values at the edges before starting the algorithm. The algorithm is stopped when a SNR of 30 dB is reached.

Machine We use a laptop based on a Core 2 Duo 2GHz processor with 2GB RAM. The algorithms are run on Matlab R2008a (Matlab 7.6.0).

4.4.2 Results

Fig. 4.5 shows the SNR as a function of the number of iterations, for each signal and each dictionary. It is important to remark that the number of MP iterations is approximately equal to the number of non-zero coefficients in the approximation, because MP may select the same atom several times, but with a low probability. By giving the SNR in function of the $l0$-norm of the approximation, these results thus compare the sparseness of the different approximations. Note that the corresponding results obtained by coding these approximations are in the next Chapter.

These results first show that increasing the size of the dictionary gives a sparser approximation i.e. for the same SNR, the approximation in a larger dictionary has smaller $l0$-norm or equivalently for the same $l0$-norm, the approximation in a larger dictionary gives a higher SNR. We have remarked that the main contribution in the SNR gain comes from the MDCT bases with the largest windows. The results also show that the performance appears to be bounded. Except for the glockenspiel signal, the dictionary 12xMDCT gives only slightly better performance than the dictionary 8xMDCT. As compared to the 8xMDCT dictionary, the dictionary 12xMDCT adds very large and very small window sizes MDCT: 32768 samples (0.74 s), 65536 samples (1.48 s), 64 samples (1.45 ms) and 32 samples (0.7 ms). These very large window sizes or very short window sizes seem to have either not enough time resolution, either not enough frequency resolution to efficiently model components of the audio signals. Finally, these results show that the performance is clearly signal dependant. The extreme cases are the glockenspiel signal and the pop music signal. With the 8xMDCT dictionary, the glockenspiel needs around 4500 iterations to reach $30dB$, whereas the pop music signal needs around 12000 iterations to reach the same SNR. Moreover, for the glockenspiel signal at 4000 iterations, the gain over the 1xMDCT dictionary is around 8 dB, whereas for the pop music signal at 12000 iterations, the gain is around 4 dB.

To better illustrate the excellent performance obtained with the glockenspiel signal, we have plotted the 30-dB approximation using two dictionaries: a single MDCT basis (1xMDCT) and a union of 8 MDCT bases (8xMDCT). Each atom is represented as a rectangle whose width is proportional to the window size and the height is inversely proportional to the window size. The colour of each atom is dependent on its energy. The plots are in Fig. 4.6. We see that the glocksenspiel signal is badly modeled by the single MDCT basis, the transients and the steady state part are modeled by a large number of atoms. Using an overcomplete basis allows to efficiently model the transients with a few short window atoms and the stationary parts with a few long window sizes atoms. This figure also raises an issue: MP creates unwanted energy before the attacks, resulting in pre-echo. This issue will be discussed in the next section.

Finally, Figure 4.7 shows computation times in function of the target SNR for the four dictionaries and two signals: gloc, popm. As the signal popm needs much more iterations to reach

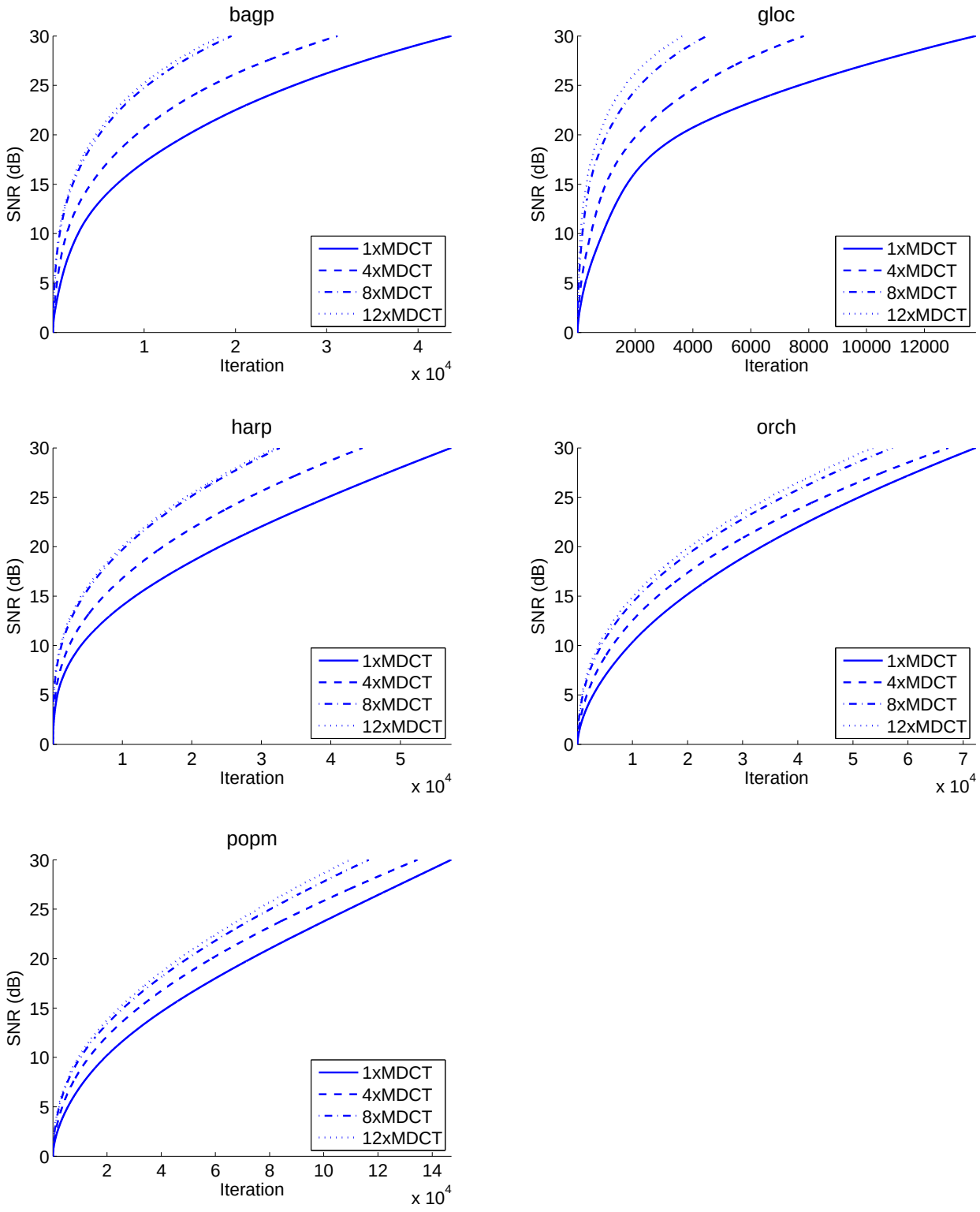

Figure 4.5: *SNR in function of the number of iterations using MP and several dictionaries.*

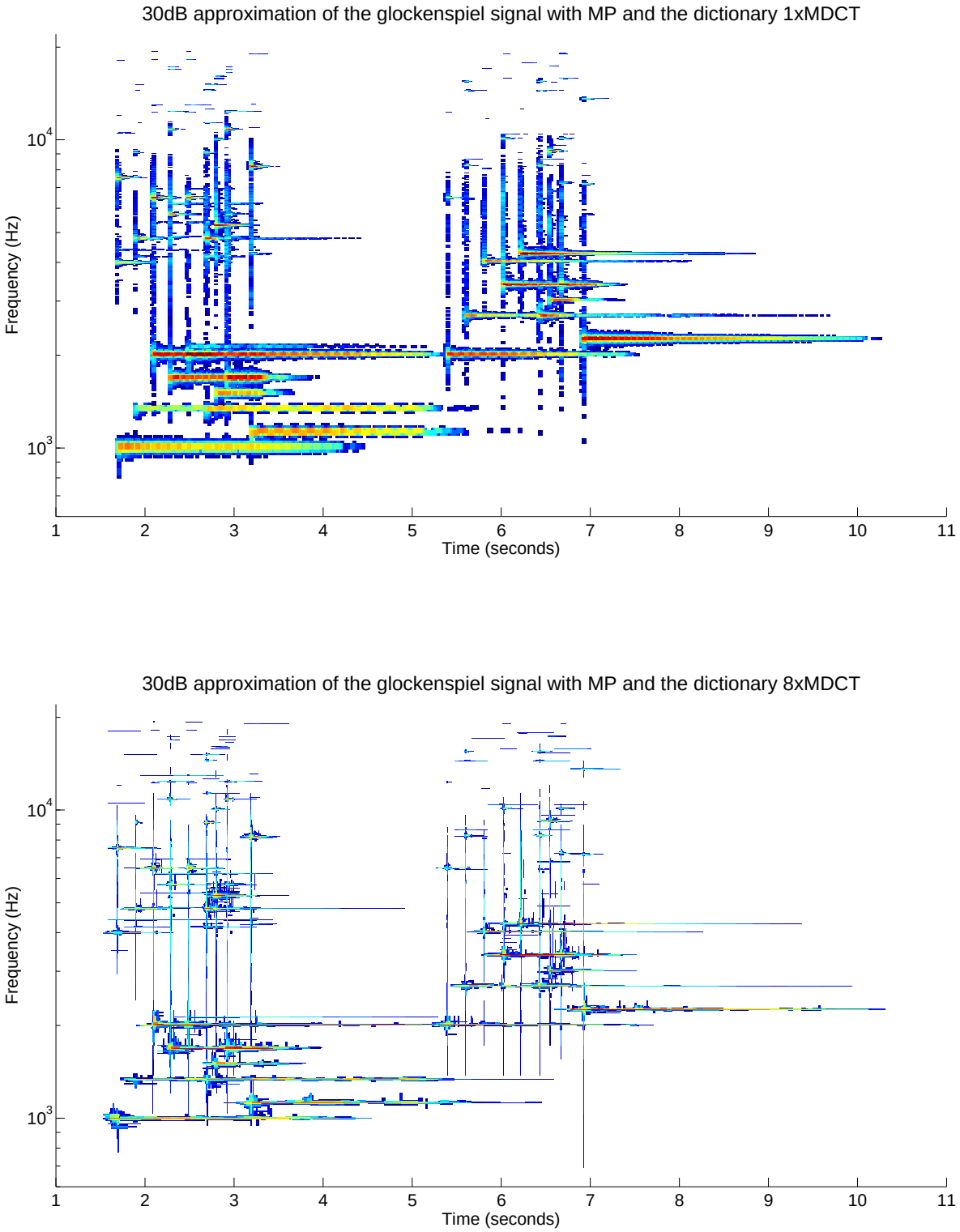

Figure 4.6: *Time-frequency plots of the 30dB approximation of the glockenspiel signal using two dictionaries: 1xMDCT and 8xMDCT.*

the same SNR as gloc, it is obvious that it needs also much more computation time. This figure shows that increasing the size of the dictionary also significantly increases the computation time. This is due to the MDCT with large window size.

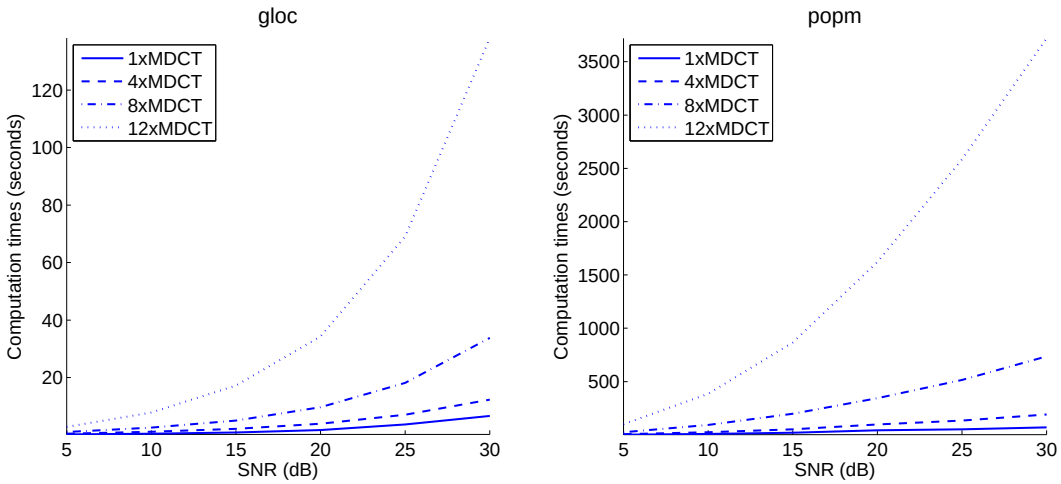

Figure 4.7: *Computation time for several dictionaries and two signals.*

4.4.3 Discussion

In this section, we have run a few experiments using Matching Pursuit in a union of MDCT bases. We have seen that the results are clearly signal dependant. Well-structured signals such as the glockenspiel signal allow very sparse approximation as the stationary parts are modeled with few long window size atoms, and the attacks are modeled with few short window size atoms. However, for less structured signals such as pop music, this is much less obvious. We have also seen that a 12-times overcomplete dictionary gives only a slight improvement compared to the 8 times overcomplete dictionary, and with much higher computational complexity. This is the main reason for our choice to use a 8 times overcomplete dictionary in the rest of this thesis. It is also interesting to remark that we have tried other dictionaries, such as a generalized MDCT where an arbitrary hop size and frequency resolution are allowed, but experimental results showed poor performance as compared to a simple union of MDCT bases (see e.g. [RD06]).

4.5 Modified Matching Pursuit with pre-echo control

4.5.1 Problem statement

Standard MP gives good results with most signals; however, it inevitably introduces pre-echo when decomposing signals containing strong attacks. This problem is illustrated in Fig. 4.8. An extract of a glockenspiel signal is decomposed with MP over a union of $M = 4$ MDCT bases (and $L_0 = 256$ samples). The second subplot shows the residual at iteration 10. The atom which is best correlated with this residual is in the third subplot. The logarithm of the absolute value of the correlation of the un-windowed function with the original signal on subframes of size 256 is in the fourth subplot. This shows that the beginning of the atom is not correlated with the signal. This results in creating energy just before the transient, which appears in the residual at iteration 11

in the fifth subplot. This energy is removed in further iterations with atoms of low energy. When coding such a decomposition at low bitrate, only the greatest energy atoms are kept which then introduces a pre-echo artifact.

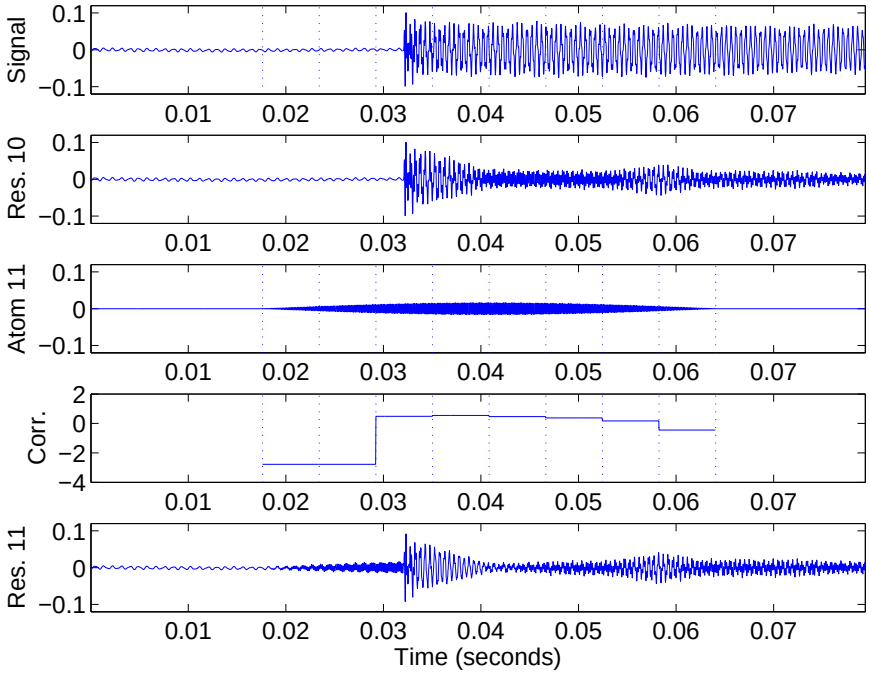

Figure 4.8: *Illustration of the pre-echo artifact with standard MP. Dotted lines correspond to subframes.*

Gribonval pointed out this problem in [GBM+96] with MP and a Gabor dictionary. He proposed a modified MP algorithm called High-Resolution Matching Pursuit (HRMP) based on the work of Jaggi et al. [JCMW95]. However, this algorithm was designed for a complex Gabor dictionary and is not adapted for a union of real MDCT bases. Moreover, HRMP significantly increases the computational cost. Alternatively, we propose a simple modification of the MP algorithm that reduces pre-echo artifacts with a small additional computational cost.

4.5.2 Proposed algorithm

At each iteration, an atom is first selected using the same criterion as MP. Then the selected atom is unwindowed and segmented in subframes as shown in Fig. 4.8. The subframes have a length which is a proportion of the atom length. We found experimentally that a good value for the number of subframes is 8. Moreover, the subframes are imposed to have a mininum length which corresponds to the length of the smallest atom in the dictionary. Then, the cross-correlation of each subframe with the corresponding segment of the original signal is computed, as shown on the 4th plot of Fig. 4.8. Finally, if the dynamic of the cross-correlations (computed as the ratio of the cross-correlation extrema) is greater than a predefined threshold, it means that the selected atom adds pre-echo and this atom is then not selected and removed from the dictionary; otherwise the atom is kept and subtracted from the residual as in MP. A pseudo-code of the modified MP

algorithm is given below.

Algorithm 9: Modified Matching Pursuit with pre-echo control

Input: The signal $\mathbf{x}$, the dictionary $\mathbf{\Phi}$ and a parameter threshold *thresh*

Output: The vector of coefficients $\mathbf{c}$ and the residual $\mathbf{r}$ such that $\mathbf{x} = \Phi\mathbf{c} + \mathbf{r}$

1 Initialization: $\mathbf{r} = \mathbf{x}$, $\mathbf{c} = \mathbf{0}$;
2 Shortest window size: W_{min} **repeat**
3 Inner products: $\mathbf{a} = \mathbf{\Phi}^T\mathbf{r}$;
4 Find maximum: $k_{max} = \underset{1 \leq k \leq K}{\operatorname{argmax}} |a_k|$;
5 Subframe length: $W = \max(\mathbf{\Phi}_{k_{max}}\text{window size}/8, W_{min})$;
6 Segment atom in subframes: $S^i\mathbf{\Phi}_{k_{max}} = i-$th W-length subframe of $\mathbf{\Phi}_{k_{max}}$;
7 Correlations of subframes with signal: $d_i = \left| < S^i\mathbf{x}, S^i\mathbf{\Phi}_{k_{max}} > \right|$;
8 **if** $max_i(d_i) >= thresh * min_i(d_i)$ **then**
9 $\mathbf{\Phi}_{k_{max}} = \mathbf{0}$;
10 Go back to step 3;
11 **end**
12 Update coefficients: $c_{k_{max}} = c_{k_{max}} + a_{k_{max}}$;
13 Update residual: $\mathbf{r} = \mathbf{r} - a_{k_{max}}\mathbf{\Phi}_{k_{max}}$;
14 **until** *a stopping condition is met* ;

4.5.3 Results

In a first experiment, we approximate an extract of the glockenspiel signal using a union of 8 MDCT bases. We compare the standard MP algorithm with the modified MP algorithms and several values for the threshold parameter. The algorithms are stopped when a SNR of 30 dB is reached. We plot a time-frequency representation as we have done in the previous section. Results are in Fig. 4.9. These results show that decreasing the threshold parameter reduce the pre-echo artifacts but also adds unwanted small window sizes atoms. A high value means that very few atoms are removed from the dictionary and thus we obtain almost the same result as with the standard MP. A low value means that a lot of atoms are removed from the dictionary, consequently the dictionary size decreases and the sparseness of the representation decreases too, with many unwanted small window sizes atoms. We found that the value 100 is a good compromise between the pre-echo reduction and the approximation sparseness.

In a second experiment, we study the Dark Energy of the approximation for several values of the parameter threshold. Dark Energy (DE) have been proposed by Sturm et al. in [SSDR08]. The authors have remarked than the MP algorithm sometimes select an atom that has high energy in a region of the signal that has no energy. This is the same phenomenom than the pre-echo artifact we have presented in this section. They have also remarked than if MP selects such an atom then it will select other atoms in the next iterations that will destructively interfere in order to preserve the original waveform. The destructive and constructive interferences between the atoms of a MP decomposition is refered as DE. DE is defined as follows

$$\Xi(n) = |\Delta(n)| \tag{4.13}$$

where $\Delta(n)$ is the difference between the energy of the approximant at iteration n and the energy that would result if the selected atom is orthogonal to the approximant at iteration $n-1$

$$\Delta(n) = \|\hat{\mathbf{x}}(n)\|_2^2 - \left(\|\hat{\mathbf{x}}(n-1)\|_2^2 + |a_{n-1}|^2\right) \tag{4.14}$$

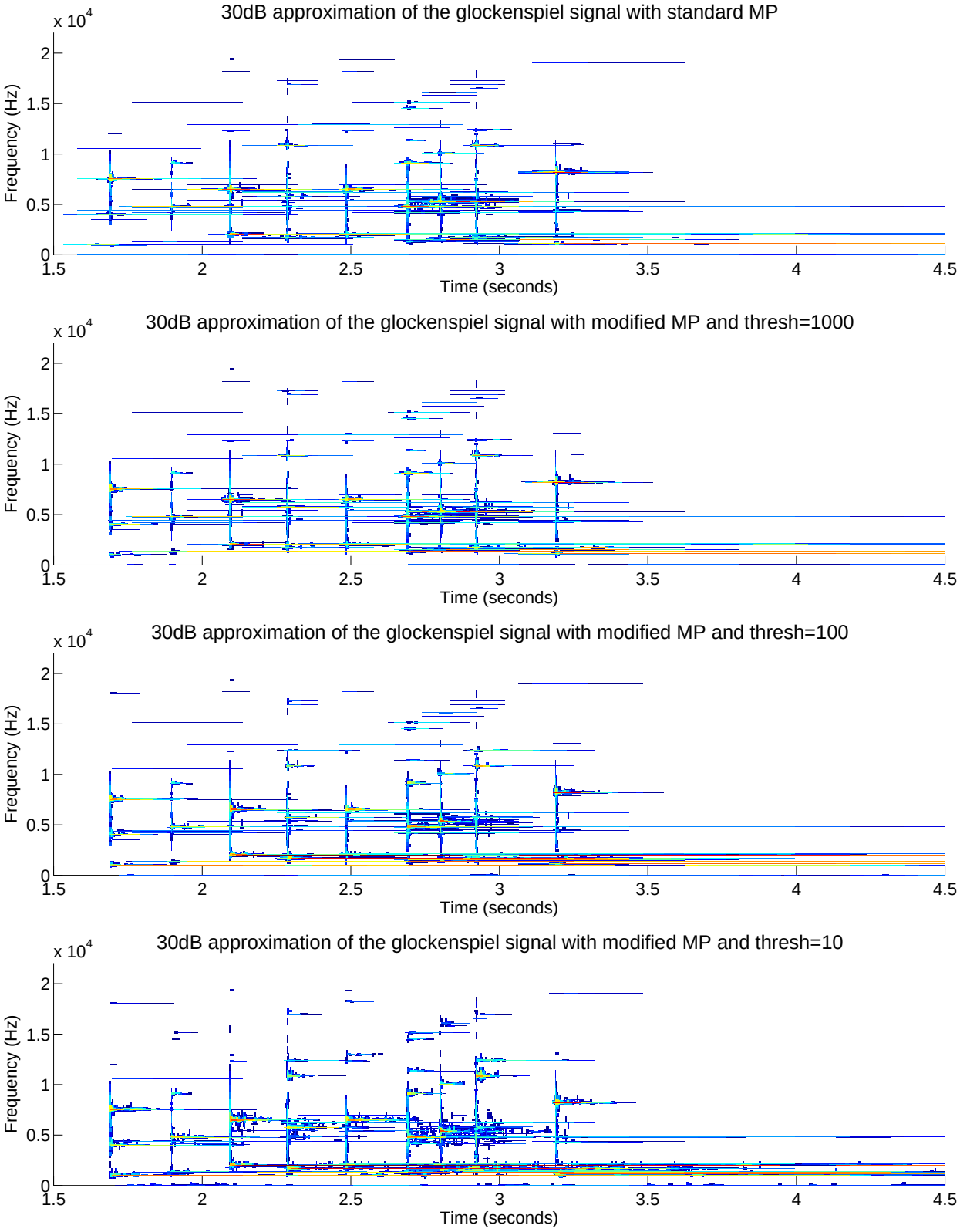

Figure 4.9: *Time-frequency plots.*

where a_{n-1} is the coefficient of the atom selected at iteration $n-1$. Fig. 4.10 plots the Cumulative Dark Energy Ratio (CDER) defined as

$$CDE_{ratio}(n) = \left(\sum_{i=1}^{n} \Xi(n) \right) / \|\mathbf{x}\|_2^2 \tag{4.15}$$

in function of the SNR. The results show that lower CDER is obtained using the modified MP with a threshold value of 100. This confirms the results obtained in the first experiment.

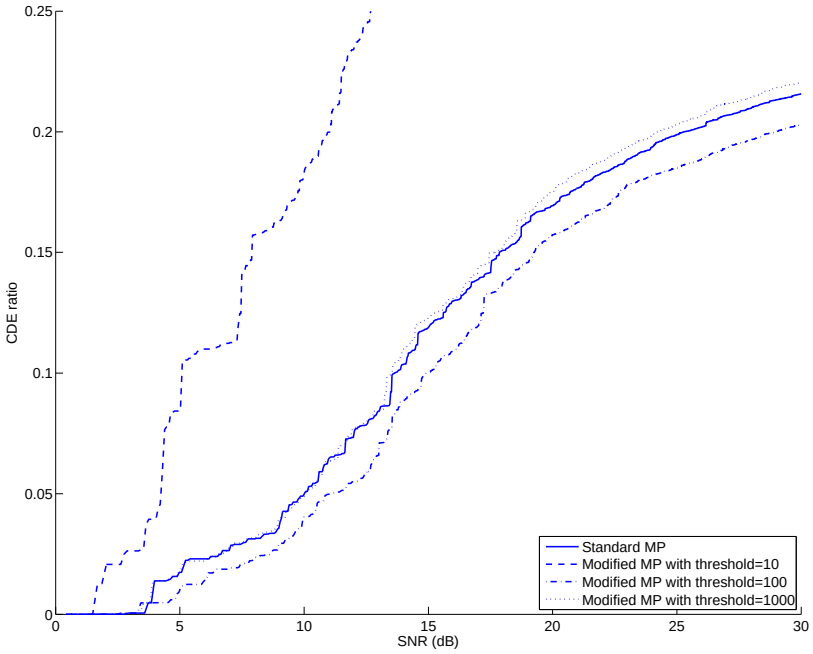

Figure 4.10: *Cumulative Dark Energy ratio for several values of the threshold parameter.*

4.6 Conclusions

We have proposed in this chapter a new approach on signal representation for audio coding. This new approach could be seen as a generalization of the traditional MDCT transform approach since it is based on a simultaneous use of a union of MDCT bases with different window sizes. We have seen that increasing the number of MDCT bases gives sparser approximations, it means that a signal is approximated by a smaller number of non-zero coefficients under a SNR constraint. We have also seen that computational complexity is a critical issue of such approximations and we have then proposed a fast algorithm for the decomposition of signals in a union of MDCT bases. Finally, we have seen that the sparsest approximation is not necessarily the best approximation from a perceptual point of view. Indeed, MP in a union of MDCT bases introduces pre-echo when decomposing signals with strong-attacks. We have then proposed a modified MP algorithm that is able to reduce the pre-echo artefact. It should be noted that this algorithm has been evaluated with the glockenspiel signal only, further experiments with other signals are still required.

In the next chapter, we will see how this new signal representation approach is applied to audio coding.

Chapter 5

Union of MDCT bases for audio coding

Abstract

This chapter brings in a new audio codec based on the signal representation approach presented in the previous chapter. The proposed audio codec is compared with a state-of-the-art pure-transform audio codec using both an objective and a subjective evaluation. Results are very signal dependant and show that much better performance is obtained with monophonic signals and similar or slightly worse performance with polyphonic signals. Results presented in this chapter have been published in IEEE Transactions on Audio, Speech and Language Processing [RRD08c].

Contents

5.1 Introduction

In the previous chapter, we have proposed a new signal representation for audio coding, based on a union of several MDCT bases with different window sizes. The signal model is written as

$$\mathbf{x} = \mathbf{\Phi c} + \mathbf{r} \tag{5.1}$$

where $\mathbf{x}$ is the signal, $\mathbf{\Phi}$ is the synthesis matrix (the dictionary), $\mathbf{c}$ the vector of coefficients (the approximation) and $\mathbf{r}$ the residual. We have then proposed algorithms that find the coefficient vector $\mathbf{c}$. In this chapter, we study how to use this signal representation for audio coding, in other words, how to encode the coefficient vector $\mathbf{c}$.

We have seen in the previous chapter that using an overcomplete union of MDCT bases allows a sparser approximation than to the traditional transform approach where a single MDCT is used. This means that for the same SNR, the coefficients vector $\mathbf{c}$ has fewer non-zero values when using an overcomplete union of MDCT bases than when using a single MDCT. However, increasing the number of MDCT bases also increases the size of the vector $\mathbf{c}$, and thus increases the number of coefficients to encode. For a low SNR, the approximation has very few non-zero coefficients, and in this case, an entropy algorithm allows a very low coding cost, as the large proportion of zero values are encoded using few bits. On the other hand, for a high SNR, the approximation has a much higher number of non-zero values, and in this case, the required coding cost is then much higher. To better illustrate this trade-off, we have approximated the 5 signals used in the experiment of the previous chapter using the standard MP with a target SNR of 100 dB and the same dictionaries as in the previous experiment: 1xMDCT, 4xMDCT, 8xMDCT and 12xMDCT. The coefficients are quantized using a simple midtread quantizer with a cell number $N = 2^k - 1, k = 2, .., 16$ and a cell width $\Delta = A/2N$ with A is the maximum of the absolute value of the coefficients. For each value of k, a bitrate is estimated using the entropy of the output of the quantizer encoder, and a SNR is computed using the decoded signal. Results are in figure 5.1. This shows that for a low SNR, lower bitrate estimates are obtained with a larger number of MDCT bases, and for a high SNR, lower bitrate estimates are obtained with a single MDCT. However, it also shows that a small number of MDCT bases never gives best performance, there is thus no compromise between a highly overcomplete union of MDCT bases and a single MDCT at medium SNR. Finally these results show that there is potentially little gain in using the dictionary 12xMDCT (except for the glockenspiel signal), this confirms the remark we have made in the previous chapter, and consequently we choose in this chapter to only use the dictionary 8xMDCT.

The remainder of this chapter is as follows. We first described the proposed source coding algorithm that we use to encode the coefficient vector $\mathbf{c}$ in sections 5.2 and 5.3. Then, we evalute the proposed audio codec in section 5.4.

5.2 Grouping and interleaving

We have formalized in the previous chapter the signal model, it is written as

$$\mathbf{x} = \mathbf{\Phi c} + \mathbf{r} \tag{5.2}$$

where $\mathbf{x}$ is the signal, $\mathbf{\Phi}$ is the synthesis matrix (the dictionary), $\mathbf{c}$ the vector of coefficients (the approximation) and $\mathbf{r}$ the residual. To simplify the notations, a coefficient is indexed using three parameters, such that we have

$$c_{m,p,k} = c_{mN+pK_m+k} \ , \ 0 \leq m < M \ , \ 0 \leq p < P_m \ , \ 0 \leq k < K_m \tag{5.3}$$

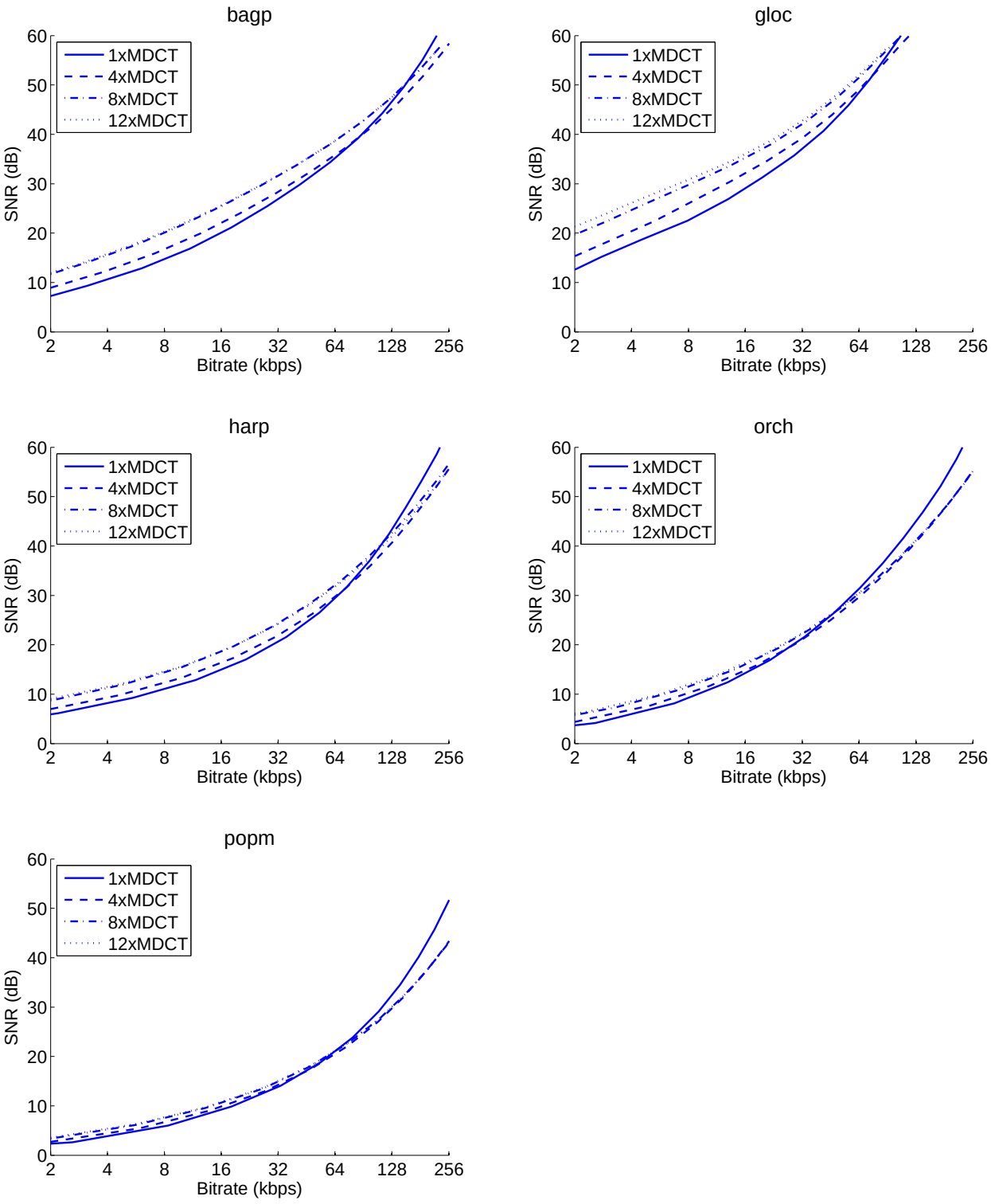

Figure 5.1: *SNR as a function of the estimated bitrate using a simple midtread quantizer.*

with m is the block index, p is the frame index, and k is the frequency index. Contrary to the transform-coding case where the analysis is done on a frame-by-frame basis, the decomposition is here performed on the whole signal. As the coefficients are encoded using similar techniques as used in transform coding, it is necessary to group the coefficients in time segments similar to the "frames" of the transform coding. These segments are called here "timeslots" (see Fig. 5.2). In each timeslot, the coefficients are interleaved to produce a vector of coefficients, which is then encoded using a bitplane algorithm described in the next section. We describe here how the coefficients are grouped in timeslots and then interleaved to produce a vector of coefficients per timeslot.

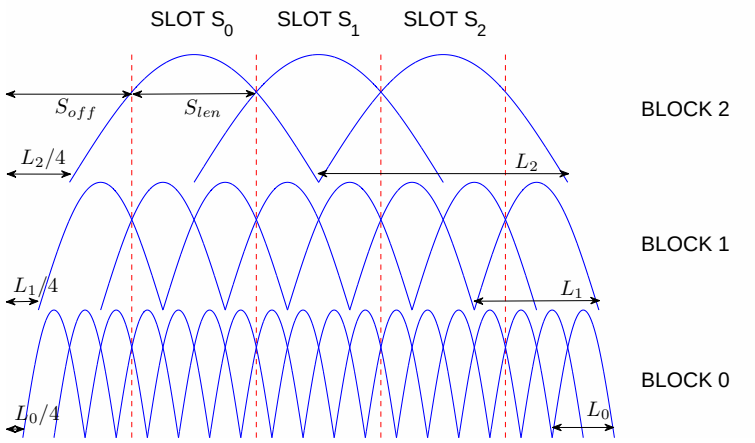

Figure 5.2: *Analysis windows for* 3 *MDCT (without the edge atoms) and corresponding timeslots.*

5.2.1 Segmentation in timeslots

The coefficients are grouped in subsets $\mathcal{S}_q$ called "timeslots", each of which includes coefficients $c_{m,p,k}$ such that the centers of the corresponding atoms are in the time support of the timeslot:

$$qS_{\text{len}} + S_{\text{off}} \leq (p+1)L_m/2 + L_m/4 < (q+1)S_{\text{len}} + S_{\text{off}} \tag{5.4}$$

where S_{len} is the timeslot length and S_{off} is the timeslot offset (position of the first timeslot). The values are chosen such that the timeslots are "aligned" with the maximum window length block (see Fig. 5.2):

$$S_{\text{len}} = S_{\text{off}} = L_{M-1}/2 \tag{5.5}$$

Using these values, the first $P'_m - 1 = 2^{M-m-1} - 1$ frames of block m are discarded and there are P'_m frames of block m in each timeslot. Timeslots are then simply defined as:

$$\mathcal{S}_q = \left\{ c_{m,p,k} \mid \text{floor}\left(\frac{p - (P'_m - 1)}{P'_m} \right) = q \right\} \tag{5.6}$$

where $\text{floor}(x)$ is the function that rounds x to the nearest integer less than or equal to x. To simplify notations in the following, we introduce a new frame index p' such that the frame index starts at 0 in each timeslot. It is defined as:

$$p' = \text{mod}\left(p - P'_m + 1, P'_m\right) \tag{5.7}$$

where $\text{mod}(x, y)$ is the remainder of the Euclidean division of x by y.

5.2.2 Coefficients interleaving

To be encoded efficiently with the runlength-based bitplane encoder described in the next section, the coefficients are interleaved so that the coefficients that are close in the time-frequency plane are put side by side. The interleaving process produces a vector of coefficients $\mathbf{v} = \{v_i | i = 1, ..., ML_{M-1}\}$.

Fig. 5.3 shows the interleaving process for a simple example where $M = 3$ and $L_0 = 2$. Coefficients are indexed using the notation XYZ where $X = m$, $Y = p'$ and $Z = k$. In 1), each row corresponds to one block and in each block, coefficients are grouped in frames. In 2) the frames of smallest scale (block 0) are interleaved two by two with the immediate upper frame in block 1. This first step produces two new frames of interleaved coefficients. In 3), these two frames are interleaved with the frame of largest scale (block 2) in a way such that the resulting vector has alternatively a coefficient of each block: one of block 2, followed by one of block 1, followed by one of block 0, followed by one of block 2, and so on.

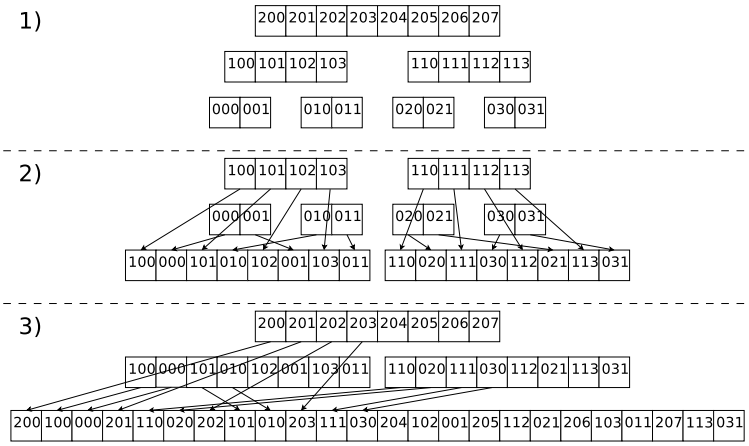

Figure 5.3: *Interleaving process for $M = 3$, $L_0 = 2$. 1) The coefficients of the three blocks are given frame by frame and block by block (XYZ is the coefficient of index $m = X$, $p' = Y$ and $k = Z$). 2) The frames of block 0 are interleaved two by two with the immediate upper frame of block 1. 3) The resulting two frames of interleaved coefficients are interleaved with the unique frame of block 2.*

The mapping process between the coefficients of a timeslot $c_{m,p',k}$ and the corresponding vector values v_i may also be formulated as follows. First we define a recursive function r that performs a permutation of the frames:

$$r(p', M-1) = p' \tag{5.8}$$

and for $m < M - 1$

$$r(p', m) = \begin{cases} r(\frac{p'}{2}, m+1) & \text{if } p' \text{ is even} \\ r(\frac{p'-1}{2}, m+1) + P'_{m+1} & \text{if } p' \text{ is odd} \end{cases} \tag{5.9}$$

Then, values are mapped according to:

$$v_i = \alpha_{m,r(p',m),k} \tag{5.10}$$

with

$$i = (kP'_m + p')M + m \tag{5.11}$$

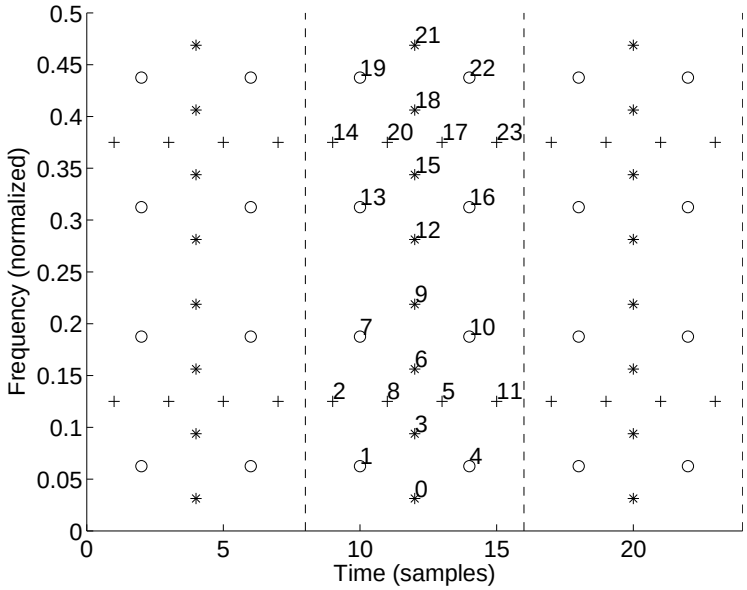

Figure 5.4: *Interleaving in the time-frequency plane. Dotted lines correspond to timeslots. Number correspond to the indexes of the vector of interleaved coefficients.*

5.3 Bitplane coding

The vector **v** of interleaved coefficients of each timeslot is encoded using bitplane encoding approaches that are similar to those used in transform coding. Though the vector length is much greater (M times) than it would be in the transform coding case, many coefficients are zero and energy is concentrated in fewer coefficients than in the transform case. Moreover, the interleaving process often clusters coefficients of high amplitudes and leaves long series of zeros as explained in the previous section. Consequently, runlength-based encoding techniques are very efficient in this case as the long series of zeros are coded using few bits. The runlength based bitplane encoder we use is based on an approach originally proposed in [Lan83]. We first describe the simple bitplane encoder as introduced in the first chapter. This algorithm is the same as the one used in some wavelet-based image coders [OWS98, Mal99a] and also in a transform-based audio coder [Dun06]. Then, we present a modified version of the bitplane encoder that shapes the quantization noise according to a psychoacoustic model.

5.3.1 Simple bitplane encoder

The coefficients v_i are first normalized by the amplitude of the coefficient with maximum amplitude $A = max(abs(v_i))$. The value A is quantized and transmitted. The coefficients are then represented in sign-amplitude form (as shown in Fig. 5.5, only the five most significant bits are shown). The j-th most significant bit of the coefficient v_i is given by $b_{i,j} = mod(floor(abs(v_i) * 2^j/A), 2)$. The vector of bits of same significance (or level) j is the j-th bitplane $B_j = \{b_{i,j}\}$. A coefficient v_i is said to be significant at level j if $abs(v_i)/A \geq 2^{-j}$. The significance of each coefficient is stored in a vector z_i ($z_i = 1$ if the coefficient is significant, $z_i = 0$ if not). Then, as explained in the first chapter, the simple bitplane encoder sends successively each bitplane starting from the most significant bitplane. This is done using a scheme in two passes. The significant pass

transmits the bits of the non already significant coefficients (BS) and the sign of the new significant coefficients. The refinement pass transmits the bits of the already significant coefficients.

For the significance pass, we use an approach based on adaptive Golomb codes [Lan83]. Here, the significance pass does not transmit directly the bits in BS but instead transmits the number of zeros between ones using adaptive Golomb codes. The parameter k of the Golomb coder is initialized to a fixed value k_{init} before encoding each bitplane. Then, the bits are encoded using the following simple algorithm: if a sequence of 2^k zeros is found in BS, a bit 0 is transmitted and k is updated $k \leftarrow k+1$; if not, it means that there remains a number of zeros inferior to 2^k followed by a one, this number of zeros is transmitted on k bits preceded by the bit 1 and k is updated $k \leftarrow k-1$. Each one found in BS corresponds to a new significant coefficient, consequently the sign of this coefficient is also transmitted. This process is repeated until the end of BS is reached. The complete algorithm of the simple bitplane encoder is detailed in Alg. 10 and 11.

Algorithm 10: Bitplane encoding

Input: A vector of interleaved coefficients $\mathbf{v} = \{v_i | i = 1...ML_{M-1}\}$
Output: The bitstream

Quantize and code max amplitude $A = max(abs(v_i))$;
Initialization: $\mathbf{z} = \mathbf{0}$, $\nu = 1$;
repeat
    Compute bitplane: $b_i = mod(floor(abs(v_i) * 2^{\nu}/A), 2)$ for all i;
    Significance pass: code $BS = \{b_i | z_i = 0\}$ and signs;
    Refinment pass: code $BR = \{b_i | z_i = 1\}$;
    Update significance: $z_i = 1$ for all i such that $b_i \in BS$ and $b_i = 1$;
    Iterate: $\nu = \nu + 1$;
until *bit budget spent or* $\nu > \nu_{max}$;

Algorithm 11: Significance pass: adaptive Golomb encoding

Input: A bitplane subset BS and a parameter k_{init}
Output: The bitstream

Initialization: $k = k_{init}$;
repeat
    if *sequence of* 2^k *zeros in* B_S **then**
        emit the bit "0";
        $k = k + 1$;
    else
        emit the bit "1";
        emit k bits: number of zeros followed by a one;
        emit 1 bit: sign of corresponding coefficient;
        $k = k - 1$;
    end
    move to the next bits in BS;
until *the end of* BS ;

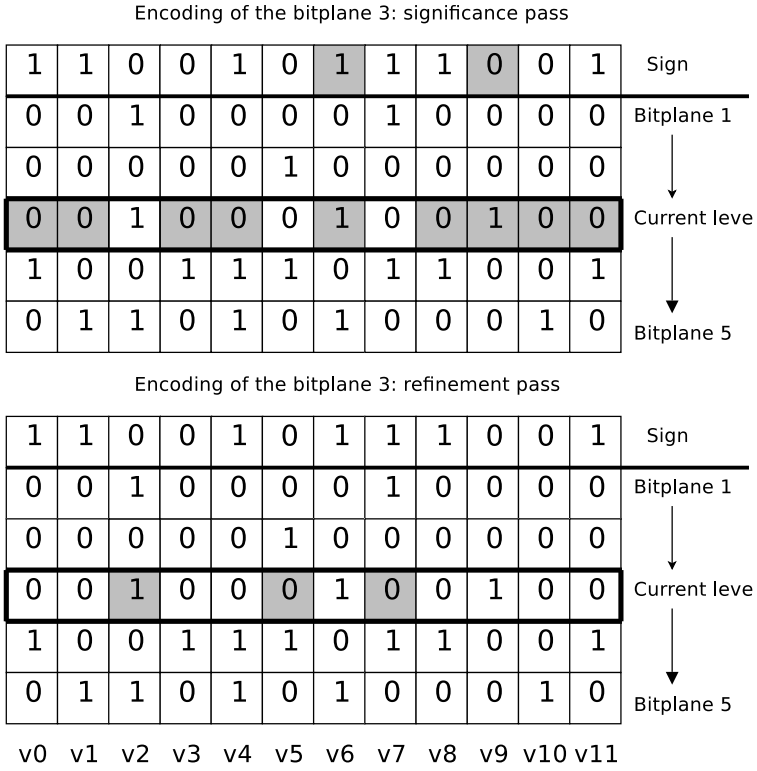

Figure 5.5: *One iteration of the simple bitplane encoder. The two encoding passes are shown. The bits in gray corresponds to the transmitted bits.*

5.3.2 Psychoacoustic bitplane encoder

The simple bitplane encoder sends the coefficients in decreasing order of amplitude. However, the coefficients with the highest amplitude are not necessarily the most perceptually relevant coefficients. Indeed some components are masked and some others are below the absolute threshold of hearing. It is therefore preferable to send first the most perceptually relevant coefficients using a psychoacoustic model. However, existing psychoacoustic approaches as used in transform coding can not be easily applied to union of MDCT representations. This is due to two main reasons.

The first reason is that the psychoacoustic models used in transform coding are designed for a fixed resolution representation and are not adapted to a multiresolution representation where time-localized components (short window atoms) and frequency-localized components (long window atoms) are superimposed. We thus propose a suboptimal approach where a masking threshold is computed for each MDCT as if they were independent MDCTs. In each frame of each block, a spectral analysis is performed and the Johnston model [Joh88] is used to compute a mask for the corresponding frame of coefficients.

The second reason is that there are more components in the overcomplete case and thus more masking values, it is then more costly to send the psychoacoustic masking threshold to the decoder. Instead we propose a suboptimal approach inspired from [Li02] where the mask is computed on the partially coded coefficients. Consequently, there is no need to transmit the mask to the decoder, as it is computed the same way by the decoder on the partially decoded coefficients.

Here, we propose a modification of the simple bitplane encoder which shapes the quan-

tization noise according to the psychoacoustic model. The complete algorithm is detailed in Alg. 12. At each iteration of the algorithm, only a subset of the current bitplane is sent. A masking threshold is used to select the bits which are transmitted. Firstly, the masking threshold is initialized to the Absolute Threshold of Hearing (ATH). Secondly, the mask is updated every time U bits have been added to the bitstream. To update the mask, a synthesized signal is first reconstructed from the partially coded coefficients. Then, a masking threshold is computed on each frame of each block as mentioned above. The psychoacoustic model gives a masking threshold value th_i for each coefficient. And finally, a simple rule decides which bits are transmitted: the difference "gap" g_i between the masking threshold (converted to the bitplane scale) and the current bitplane level is computed: $g_i = -\log2(\sqrt{th_i}/A) - \nu_i$. Then we select the bits whose gaps are greater than the mean of the gap: j such that $g_j \geq \text{mean}(g_i)$. Fig. 5.6 shows one iteration of the algorithm. Contrary to the simple bitplane encoder, the current level is different for each coefficient because only a subset of the bitplane is sent at each iteration. The gap is computed for each coefficient and the coefficients whose gap value is above the mean of the gap are selected, here the coefficients 0,1,2,8,9,10,11. Finally, the selected bits are encoded using the same two-pass scheme as in the simple bitplane encoder.

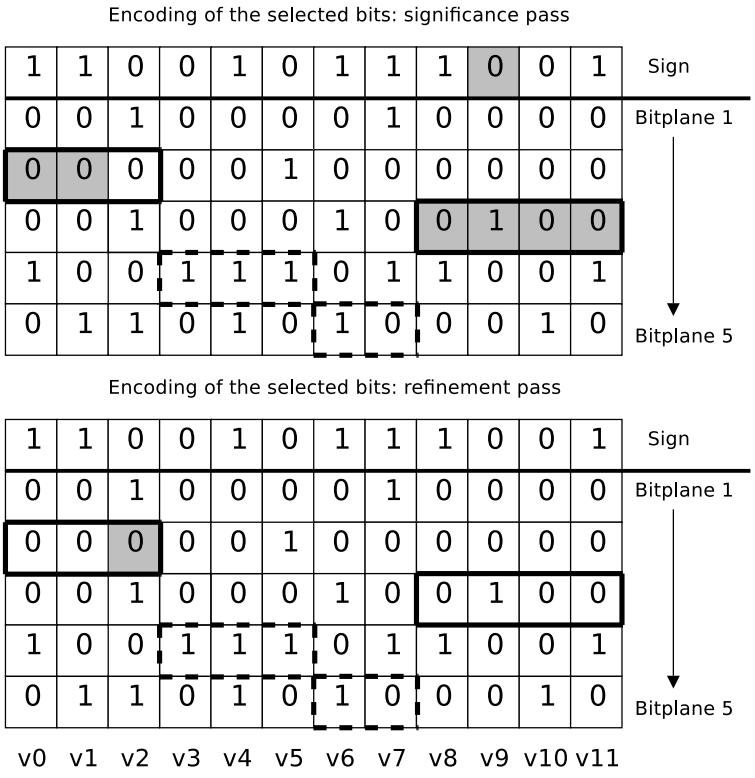

Figure 5.6: *One iteration of the psychoacoustic bitplane encoder. Plain rectangles correspond to the level of the selected coefficients. Dotted rectangles correspond to the level of the non-selected coefficients. The two encoding passes are shown. The bits in gray corresponds to the transmitted bits.*

Algorithm 12: Psychoacoustic bitplane encoding

Input: A vector of interleaved coefficients $\mathbf{v} = \{v_i | i = 1...ML_{M-1}\}$
Output: The bitstream

Quantize and code max amplitude $A = max(abs(v_i))$;
Initialization: $\mathbf{z} = \mathbf{0}$, $\nu = \mathbf{1}$;
Initialize the masking threshold **th** to the ATH;
repeat
  if *U new bits have been added to the bitstream* **then**
    update mask **th**;
  end
  Compute gap: $g_i = -log2(\sqrt{th_i}/A) - \nu_i$ for all i;
  Make selection: $s_i = g_i \geq mean(g_i)$ for all i;
  Compute bitplane: $b_i = mod(floor(abs(v_i) * 2^{\nu_i}/A), 2)$ for all i;
  Significance pass: code $BS = \{b_i | z_i = 0 \text{ and } s_i = 1\}$ and signs;
  Refinment pass: code $BR = \{b_i | z_i = 1 \text{ and } s_i = 1\}$;
  Update significance: $z_i = 1$ for all i such that $b_i \in BS$ and $b_i = 1$;
  Iterate: $\nu = \nu + 1$;
until *bit budget spent or* $\nu > \nu_{max}$;

5.4 Evaluation

We evaluate in this section the proposed audio coder. We first evaluate the source coding algorithm. We then study the influence of the pre-echo control MP on the audio coder performance. And finally, we evaluate the final codec using an objective measure and a listening test.

We use PEMO-Q software [HK06] as an objective measure to evaluate our coder. This software gives two measures, the Perceptual Similarity Measure (PSM) which is restricted to the interval $[-1, 1]$ (1 indicates transparency) and the Objective Difference Grade (ODG) which is on the same scale as the Subjective Difference Grade (0 indicates transparency). It is important to remark that PEMO-Q was optimized and validated with high bitrates audio coders. Consequently, the results at low bitrates may be less relevant.

5.4.1 Source coding algorithm

We have proposed in the previous section a source coding algorithm where the coefficient vector produced by MP is first split into subsets called timeslots, and the coefficients in each timeslot are interleaved and encoded using a bitplane algorithm. We have seen that the simple bitplane encoding algorithm depends mainly on the way the significance map is encoded. In a first experiment, we study the influence of the algorithm used to encode the significance map. The five signals listed in Annex A are first approximated using MP in the union of 8 MDCT bases. Then, the resulting coefficient vector is split in timeslots and a vector of interleaved coefficient per timeslot is produced. Each vector of coefficient is then encoded using the simple bitplane algorithm described in the previous section. For the significance map coding, we compare two algorithms: the run-length algorithm based on adaptive Golomb coding described in the previous section, and the adaptive arithmetic coding algorithm of [Sai04]. Fig. 5.7 shows the mean of the coding cost of the significance map of each bitplane and the mean of computation time needed to reach a target bitrate. The mean is computed over the five signals. Results show that the Golomb based coder gives better performance than the arithmetic based coder in the first bitplanes and slightly worse

on the last bitplanes, and at a lower computational cost. We then choose the Golomb based coder in the rest of this thesis.

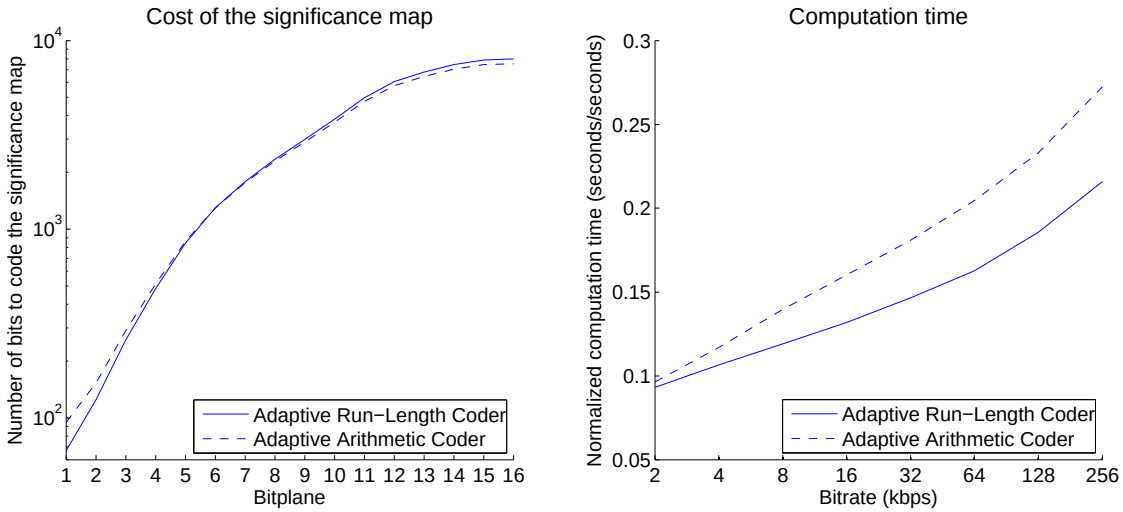

Figure 5.7: *Comparison of two algorithms for the coding of the significance map in simple bitplane coding.*

In a second experiment, we study the influence of the proposed interleaving algorithm. Fig. 5.8 shows the mean of coding cost of the significance map of each bitplane, using either the proposed algorithm or a random interleaving. If the coefficients were i.i.d, any interleaving algorithm would produce the same results. However, the coefficients are not i.i.d, and the proposed algorithm allows to reduce the coding cost of the significance map. It is important to note that other interleaving algorithms are possible, we tested several alternatives and there were no noticeable differences in most cases.

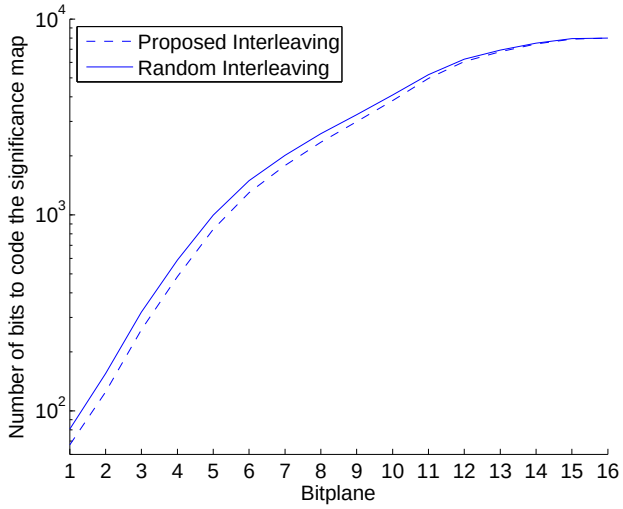

Figure 5.8: *Influence of the proposed interleaving algorithm on the coding cost of the significance map in simple bitplane coding.*

5.4.2 Modified MP with pre-echo control

We study here the influence of the modified MP with pre-echo control on the audio coder performance. A 4-seconds extract of the glockenspiel signal is approximated using either the standard MP or the modified MP, in the union of 8 MDCT bases. The coefficients are then encoded using the interleaving process and the simple bitplane encoding described previously. As a reference, the signal is represented using the same time-varying MDCT as used in AAC (long window sizes 2048 and short window sizes 256), and the resulting coefficients are encoded using the same source coding algorithm. We then compare the degradation levels introduced by these three codecs. As formal listening tests are very time consuming, we use here the PEMO-Q objective measures only. Fig. 5.9 shows the obtained PSM and ODG measures for the three codecs as a function of the target bitrate. This shows that the modified MP gives better performance than the standard MP and that proposed approach clearly outperforms the reference codec.

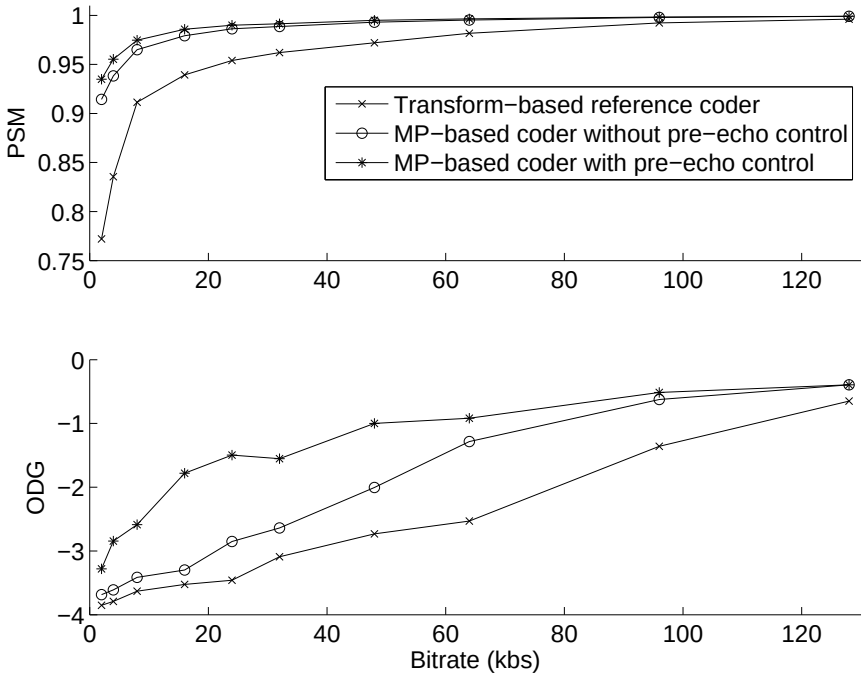

Figure 5.9: *Influence of the modified MP with pre-echo control, and comparison with a reference transform-based codec.*

5.4.3 Final codec: objective evaluation

Finally, we evaluate the final proposed codec, using the modified MP in a union of MDCT bases, and the psychoacoustic bitplane coding proposed in the previous section. The final codec use the following parameters.

1) Signal representation parameters The coder tested is based on a union of $M = 8$ bases, with the shortest window length being $L_0 = 128$. Two analysis windows have been tested: the sine window and the KBD window. Since there were no noticeable differences in the results using one or the other, we used the sine window. The signal is padded with one second of zeros at both sides. The parameter *thresh* in the modified MP is set to 100 and the target SNR is set to 80 dB. We found it necessary to use such a high value to obtain relevant objective measurements at

high bitrates (above 128 kbps). However, in practice, it is sufficient to use a target SNR between 30 and 50 dB for low to medium bitrates.

2) Source coding parameters The maximum amplitude value is quantized and coded using 16 bits. The parameter k_{init} is set to 2. The bit budget for each timeslot is constant and is set according to the target bitrate. The maximum level v_{max} is set to 30. The number of bits U necessary to update the mask is set to 1 kbits. The critical bands for the Johnston model have a length of approximately 0.3 barks. The ATH formula is the one provided in [PS00], and the spreading function formula is the one provided in MPEG-4 specifications [ISO01].

In our first experiment, our coder is compared to a similar transform-based coder. This makes it possible to assess the usefulness of using an overcomplete basis, all other things (quantizing, coding) being equal. The reference coder uses only $M = 1$ MDCT basis with $L_0 = 2048$ samples and exactly the same coding algorithm. Note that the results of the transform coder may have been improved by using a time-varying MDCT with a switching block size at transients. The mean of the PSM and ODG measures over the five signals are in Fig. 5.10. The union of MDCT approach gives significantly better results at low bitrates while performing similarly at high bitrates.

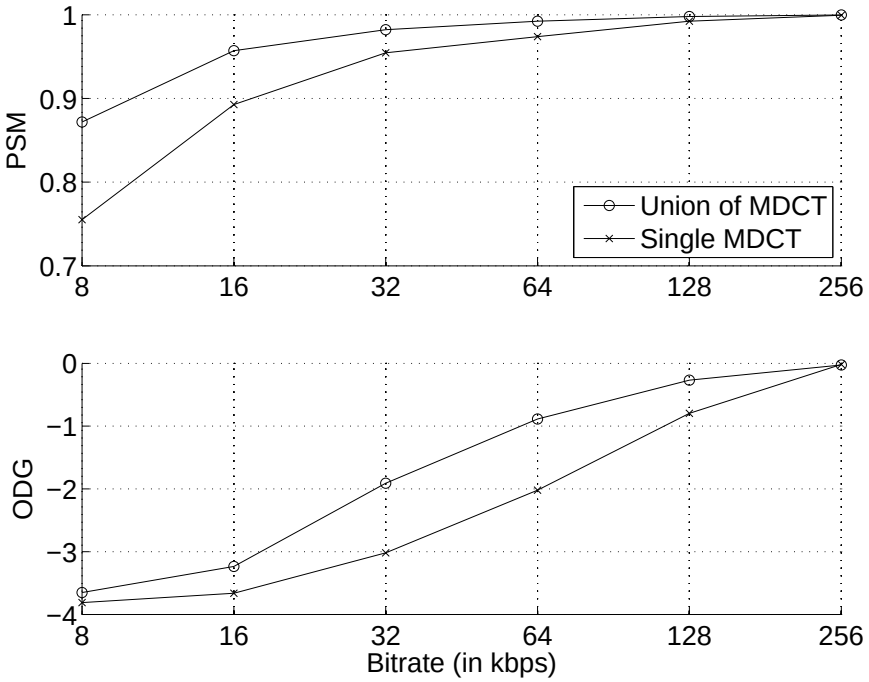

Figure 5.10: *Mean of PSM and ODG objective measures for 5 signals and 2 coders based on a single MDCT and a union of 8 MDCT*

In our second experiment, two versions of our coder are compared with a reference state-of-the-art coder. The two versions of our coder are the coder without the psychoacoustic model and the coder with the psychoacoustic model. The reference coder is the iTunes 7 AAC encoder [iTu08] as it is a freely available and fully AAC compatible encoder. The mean of the PSM and ODG over the five signals are in Fig. 5.11. The results show that the psychoacoustic model adds a significant gain to the results of the previous coder, and gives a coder which is competitive with the iTunes AAC encoder.

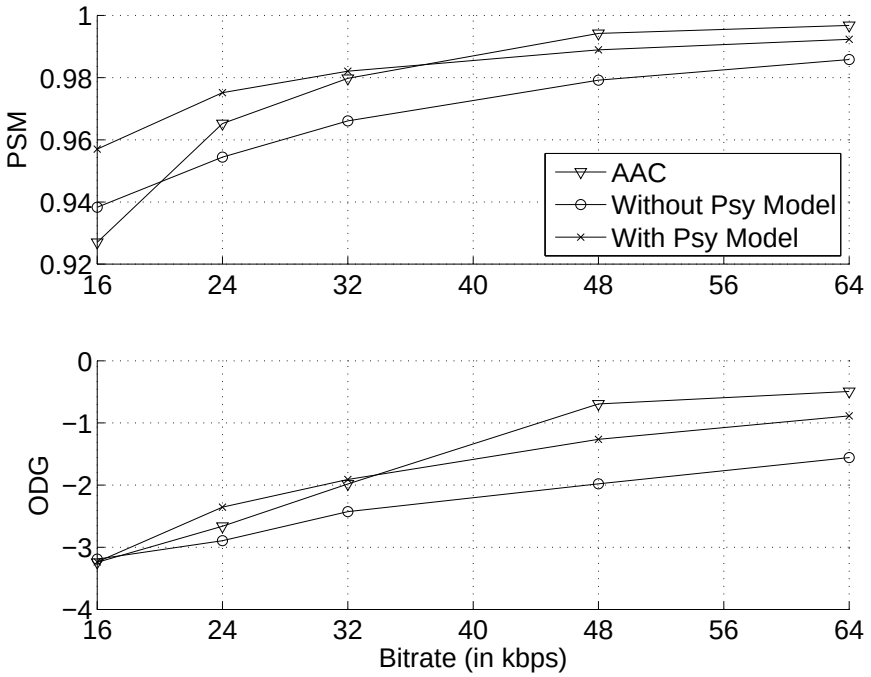

Figure 5.11: *Mean of PSM and ODG objective measures for 5 signals and 3 coders: iTunes's AAC, our coder without psychoacoustic model, our coder with psychoacoustic model.*

5.4.4 Final codec: subjective evaluation

Two versions of our coder are subjectively evaluated with a MUSHRA listening test [ITU03], with scores that range from 0 (extremely poor quality) to 100 (transparent). Ideally, an ABC test would also be needed to test transparency at high bitrates. But performing a listening test is very time consuming, and we have preferred to test the performance of our coder at medium and low bitrates only. A total of 20 listeners have evaluated 8 versions of the five test signals: a hidden reference, a 3.5 khz low-pass anchor, two signals coded with iTunes AAC at 24 and 48 kbps, two signals coded with our coder and the simple bitplane encoder (noted `codec_A`) at 24 and 48 kbps, and two signals coded with our coder and the psychoacoustic bitplane encoder (noted `codec_B`) at 24 and 48 kbps. These 40 sound files and the source code of the codec are available online[1]. The participants were post-screened according to the score they attributed to the reference: the data for listeners who gave a score under 90 (which correspond to the lowest mark for the category "excellent quality") were discarded. A total of 12 listeners were kept, mostly graduate students working in the field of acoustics or audio processing. Fig. 5.12 presents the scores for each 5 signals, with the overall scores shown at the bottom.

The first goal of this listening test is to show the benefit of the psychoacoustic bitplane encoder (`codec_B`) compared to the simple bitplane encoder (`codec_A`). The results are very signal dependant. Simple signals (bagp, harp and gloc) are very sparse in the time-frequency domain, in this case there is no gain when using the psychoacoustic bitplane encoder. For harp and gloc signals, the psychoacoustic approach even slightly degrades the performance, though scores still remain high. For more complex, polyphonic signals, the psychoacoustic approach significantly increases

[1]The sound files used in the listening tests and the GNU/GPL source code of the codec needed to reproduce the results are available online at the following address `http://www.emmanuel-ravelli.com/taslp08`

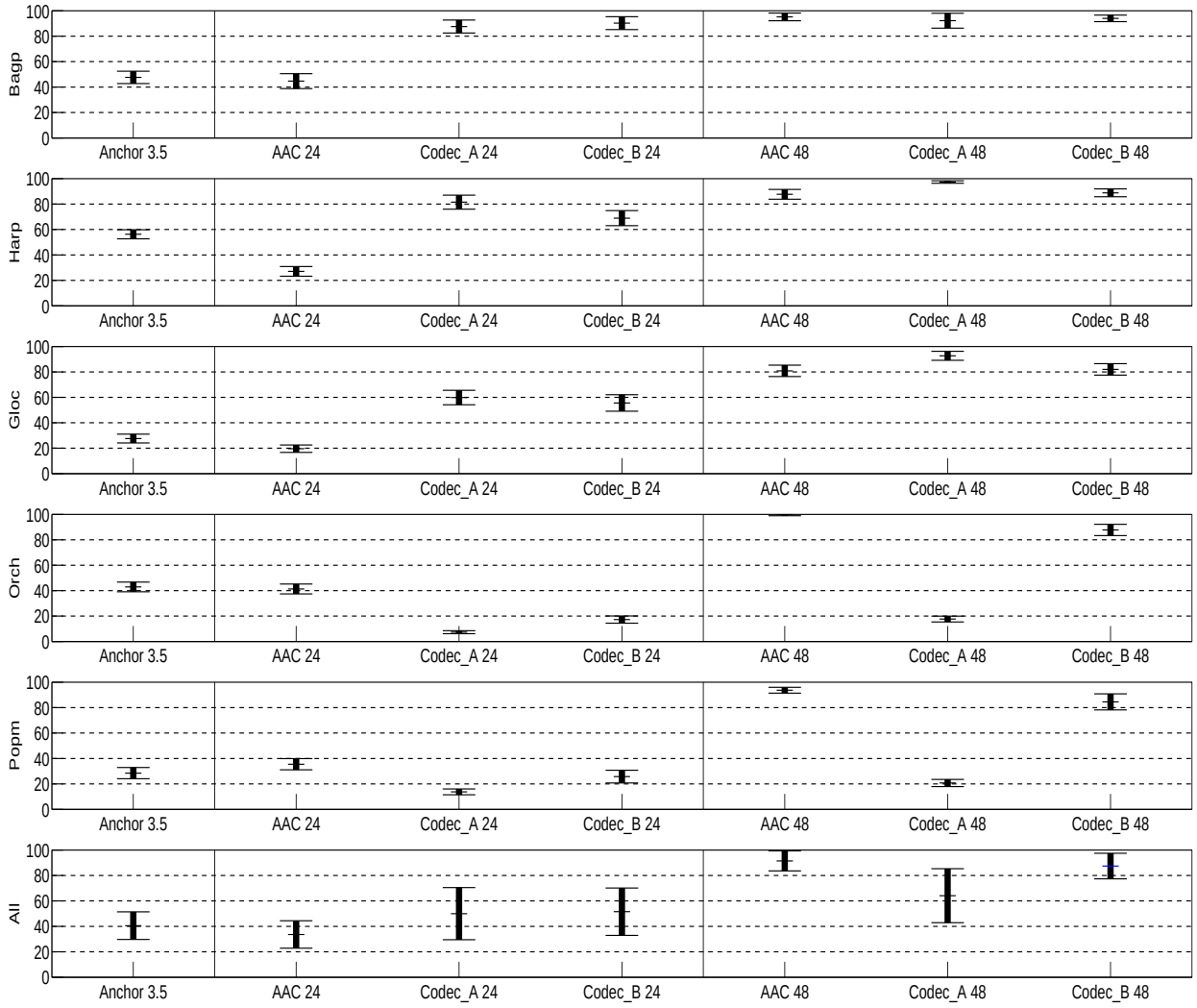

Figure 5.12: *MUSHRA listening test results. From left to right: 3.5 khz low-pass anchor, iTunes AAC at 24 kbps, our coder with the simple bitplane encoder (Codec A) at 24 kbps, our coder with the psychoacoustic bitplane encoder (Codec B) at 24 kbps, iTunes AAC at 48 kbps, our coder with the simple bitplane encoder (Codec A) at 48 kbps, our coder with the psychoacoustic bitplane encoder (Codec B) at 48 kbps. From top to bottom: Bagpipe, Glockenspiel, Harpsichord, Orchestra, Pop Music, All signals. Error bars indicate 95% confidence intervals.*

the scores. This is due to the presence of a very large number of time-frequency components that mask each other.

The second goal of this listening test was to compare our proposed coders with a reference pure transform coder, the iTunes AAC coder. These results are also very signal dependant: at low bitrates, our approach gives much better results for monophonic signals (bagp, harp and gloc) and slightly worse results for polyphonic signals (orch and popm). At high bitrates, our approach with the psychoacoustic model gives the same or slightly worse results, except for the case of polyphonic signals and the simple bitplane encoder, where the poor results show that a psychoacoustic approach is necessary.

Overall, these results show that our approach is competitive with a state-of-the-art pure transform coder, especially since we did not spend much effort in the optimization of encoding parameters as opposed to the highly-optimized AAC coders. It is also interesting to note here that, as opposed to the reference coder, our approach provides fine-grain scalability, a property which is generally provided at a cost in terms of performance. An example of fine-grain scalable coder is MPEG-4 BSAC. Formal listening tests [ISO99] show that at low bitrates BSAC requires a 12.5 % bitrate overhead compared to AAC, for the same quality.

5.5 Conclusions

We have presented in this chapter a novel audio codec based on an original representation. Contrary to existing methods, this approach provides transparency at high bitrates and competitive results at low bitrates. Listening tests that compare our proposed coder with a state-of-the-art transform-based coder show that our method gives a much better quality for monophonic signals and similar or only slightly worse results for polyphonic signals. However, it should be emphasized that the main goal was to show that the concept of sparse overcomplete representations over a union of MDCT bases is a viable approach for audio coding. Our coder has also been designed with the extra constraint of fine-grain scalability, although this is by no means a requirement of our concept.

However, these results are obtained at a cost in terms of computation time. Indeed, the decomposition algorithm is much slower than what is needed for a transform (which approximately corresponds to the first iteration of the Matching Pursuit algorithm). Moreover, our approach uses much longer analysis window lengths (up to 16384 samples at 44.1 kHz) than what is used in an AAC coder (2048 samples) which gives a much greater delay time. The delay of our coder is equal to the maximum window length i.e. 371.5 ms. These two issues prevent applications that require real-time/low-delay encoding. The computation times are high but not excessive (typically between 5 and 200 times the duration of the file, depending on the signal and the target bitrate), and are acceptable for off-line sound databases or personal music collections.

While the proposed coder is competitive with state-of-the-art transform coder, there is still much room for improvement. One possibility is to use more efficient decomposition algorithms such as orthogonal Matching Pursuit [DMA97], Basis Pursuit [CDS99] or Gradient Pursuit [BD08]. We can also improve the coding part: a better psychoacoustic model, a better integration of the psychoacoustic model into the bitplane encoder and a fixed bitrate coding scheme instead of bitplane encoding.

Chapter 6

Matching Pursuit in adaptive dictionaries for scalable audio coding

Abstract

This chapter proposes an improved matching pursuit algorithm over a union of MDCT bases for scalable audio coding. It is based on a novel switching procedure that adaptively reduces the size of the dictionary using a rate-distortion optimization. The resulting audio codec gives better SNR than the previous codec at high bitrate, and with a reduced computational complexity. The results presented in this chapter have been published in the proceedings of EUSIPCO 2008 [RRD08b].

Contents

6.1 Introduction

In the previous chapter, we have proposed a new signal representation that allows better performance than transform coding at low bitrates while allowing transparent quality at high bitrates. In comparison, for the same target SNR, there are less significant coefficients to encode than in the transform case, but encoding the parameters of the significant coefficients (the significance map) is more costly. We have shown that the tradeoff between the number of significant coefficients, and the coding cost of these coefficients, significantly favors our approach at low bitrates. However, at high bitrates, it is necessary to encode a high number of coefficients in both approaches, and the cost of encoding large significance map becomes prohibitive: in this case our approach is outperformed by the standard transform approach.

In this chapter, we propose a new decomposition algorithm that, under mild assumptions, provides the "best of both worlds": the same performance as the previous approach at low bitrates and the same performance as transform coding at high bitrates. The signal is first approximated in the overcomplete set of time-frequency atoms used in the previous chapter (union of 8 MDCT bases), and then the residual of this approximation is decomposed using an orthogonal transform (one of the MDCT bases). The signal decomposition is performed on the whole signal using a modified Matching Pursuit (MP) algorithm, with an adaptive dictionary that is changed locally i.e. the union of MDCT is reduced to one MDCT on a frame-by-frame basis. The decomposition is then encoded using similar bitplane encoding methods as used in previous chapter. The main issue is the design of an efficient strategy to decide on the appropriate MP iteration, in a given frame, to switch from the overcomplete to the complete dictionary.

A similar idea is found in image coding [PGV06], where an overcomplete set of 2D atoms is used to model the edges of an image and the residual of this approximation is coded using a wavelet transform. However, this approach is based on two different dictionaries and two different coding methods. The image coder combines this two different approaches in a rate-distortion way. Another similar approach is found in audio coding [vv05], where SSC is used to approximate a signal and the residual is coded using a MDCT-based coder. A rate-distortion optimization is used to allocate the available bit budget between the two coders. Our approach is however slightly different. We propose an alternate solution where a single paradigm for the signal representation and the coding method is used, both are adapted online using the novel switching procedure proposed in this chapter.

The remainder of this chapter is as follows. In section 6.2, we introduce the signal model and the decomposition algorithm. In section 6.3, we describe how the decomposition is encoded and derive an optimal parameter rate-distortion value for the adaptive decomposition algorithm. In section 6.4, we present the results, and finally we conclude in section 6.5.

6.2 Signal representation

6.2.1 Signal model

An audio signal is approximated using the same union of 8 MDCT bases as in the previous chapter. We recall here briefly the signal model. Considering a signal $\mathbf{x}$ of length N samples, it is approximated as

$$\mathbf{x} = \mathbf{\Phi c} + \mathbf{r} \tag{6.1}$$

with $\mathbf{r}$ is the residual, $\mathbf{c}$ the vector of coefficients of length MN and $\mathbf{\Phi}$ the synthesis matrix of size $N \times MN$ defined as:

$$\mathbf{\Phi}\left(n, mN + pK_m + k\right) = g_{m,p,k}(n) \ , \ 0 \leq m < M \ , \ 0 \leq p < P_m \ , \ 0 \leq k < K_m \ , \ 0 \leq n < N \quad (6.2)$$

where P_m is the number of segments of length $K_m = L_m/2$ such that $N = P_m K_m$, and

$$g_{m,p,k}(n) = w_{m,p}(u)\sqrt{\frac{2}{K_m}}\cos\left[\frac{\pi}{K_m}\left(u + \frac{K_m}{2} + \frac{1}{2}\right)\left(k + \frac{1}{2}\right)\right] \quad (6.3)$$

and

$$u = n - \left(p - \frac{1}{2}\right)K_m \quad (6.4)$$

The window $w_{m,p}(u)$ is a sine-based window.

6.2.2 Decomposition algorithm

In the previous chapter, the signal was decomposed using MP. Standard MP is performed globally on the whole signal. To better adapt the model to the local statistics of the signal, we consider here a modified MP algorithm that adapts the dictionary in the "timeslots" defined in the previous chapter. We first recall briefly a few definitions. The timeslots group coefficients in subsets $\mathcal{S}_q$ defined as

$$\mathcal{S}_q = \left\{ c_{m,p,k} \mid \text{floor}\left(\frac{p - (P'_m - 1)}{P'_m}\right) = q \right\} \quad (6.5)$$

where

$$c_{m,p,k} = c_{mN+pK_m+k} \ , \ 0 \leq m < M \ , \ 0 \leq p < P_m \ , \ 0 \leq k < K_m \quad (6.6)$$

are the coefficients and $P'_m = 2^{M-m-1}$ is the number of frames of block m in each timeslot. In the timeslot $\mathcal{S}_q$, the time support of the largest scale atom includes all smaller scale atoms. Consequently, we define the time support U_q of the timeslot $\mathcal{S}_q$ as the time support of the largest scale atom. At every MP iteration, one atom is picked up, that can belong to any timeslot. We thus define $n_q(i)$ the MP iterations where the selected atom belongs to the timeslot $\mathcal{S}_q$. The atom selected at iteration $n_q(i)$ decreases the energy of the residual on the time support U_q. We thus define the SNR of the timeslot $\mathcal{S}_q$ as

$$SNR_q(i) = -20\ \log_{10}\left(\frac{||R^{n_q(i)}_{U_q}||}{||R^0_{U_q}||}\right) \quad (6.7)$$

and the SNR decay of the timeslot S_q as

$$DEC_q(i) = -20\ \log_{10}\left(\frac{||R^{n_q(i)+1}_{U_q}||}{||R^{n_q(i)}_{U_q}||}\right) \quad (6.8)$$

with $R^n_{U_q}$ is the part of the residual at iteration n corresponding to the time support U_q. Fig. 6.1 plots the decay curve $DEC_q(i)$ as a function of $SNR_q(i)$, obtained with the standard MP for a timeslot of an audio signal and two dictionaries: one single MDCT with window length 2048 samples and the union of 8 MDCT bases. In the first iterations, each atom of the overcomplete dictionary decreases significantly the SNR of the timeslot, but after some iterations, the SNR decay

becomes small and almost equal to the one of the orthogonal dictionary at same SNR. This is the same phenomenon as the one described in [MZ93]: the atoms extracted in the first iterations are the coherent structures of the signal and after some iterations the residue R^n converges to a process called the dictionary noise. The SNR decay of the overcomplete dictionary even tends towards a constant which depends only on the dictionary and is equal in this case to approximately 0.002. When coding such a decomposition, there is a gain in the first iterations but after some point, it is less costly to encode an atom from an orthogonal dictionary. Consequently, from a coding point of view, it is better to decompose the first iterations in the union of MDCT bases, and then to reduce the dictionary when the decay becomes too low.

The proposed modified MP is detailed in Alg. 13. At each iteration, the atom that is most correlated with the signal is selected (as in standard MP). Then, the SNR decay in the corresponding timeslot is computed. If the SNR decay is below a constant SW (we will show later how to compute this constant), the dictionary subset corresponding to that timeslot is reduced to the orthogonal dictionary i.e. all atoms in the timeslot are removed from the dictionary except those from the orthogonal dictionary in every timeslot. One possibility is to stop the algorithm when only atoms from the orthogonal dictionary remains in the dictionary. However, we have found that the stopping criterion could be simplified as follows. The algorithm is stopped when a given number of consecutive atoms from the orthogonal dictionary are selected (for example 100). This has for consequence to stop the algorithm sooner and thus lower the complexity, and with a negligible loss in terms of performance. Finally, the residual is projected on the orthogonal dictionary and the coefficients are updated. This algorithm has been implemented in the MPTK framework using the fast implementation described previously.

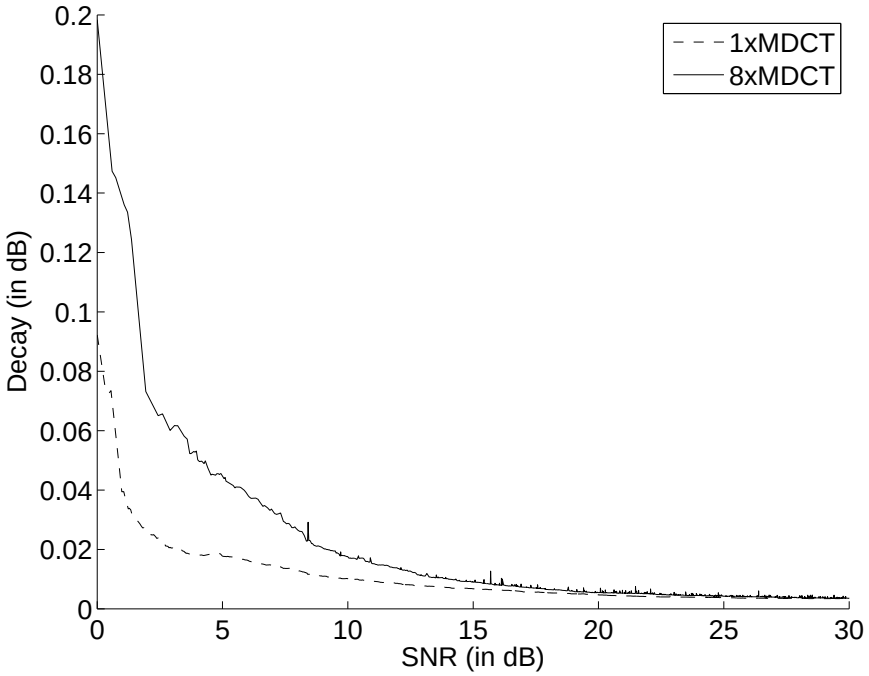

Figure 6.1: *SNR decay as a function of SNR for a timeslot*

Algorithm 13: Adaptive Matching Pursuit

Input: The signal $\mathbf{x}$, the dictionary Φ and a parameter SW

Output: The vector of coefficients $\mathbf{c}$ and the residual $\mathbf{r}$ such that $\mathbf{x} = \Phi\mathbf{c} + \mathbf{r}$

Initialization: $\mathbf{r} = \mathbf{x}$, $\mathbf{c} = \mathbf{0}$;
repeat
  Inner products: $\mathbf{a} = \Phi^T\mathbf{r}$;
  Find maximum: $k_{max} = \underset{1 \le k \le K}{\operatorname{argmax}} |a_k|$;
  Update coefficients: $c_{k_{max}} = c_{k_{max}} + a_{k_{max}}$;
  Update residual: $\mathbf{r} = \mathbf{r} - a_{k_{max}}\Phi_{k_{max}}$;
  Compute timeslot index q of the selected atom;
  if $DEC_q < SW$ **then**
    remove from Φ all atoms in timeslot q except those from the orth. dict.;
  end
until *only atoms from the orth. dict. remain in the dict.* ;
Project the residual $\mathbf{r}$ on the orth. dict. and update coefficients;

6.3 Coding

6.3.1 Adaptive bitplane coding

A vector of interleaved coefficients per timeslot is first produced using the same interleaving process as in the previous chapter. Fig. 6.2 shows the amplitude of the coefficients of such a vector on the bitplane scale. The coefficients have been normalized, sorted in decreasing order and converted on the bitplane scale using $\log_2$. The coefficients that belongs to the orthogonal dictionary are plotted using a dashed line, the other coefficients are plotted using a plain line. As expected, the coefficients in the first bitplanes (i.e. with the highest amplitudes) belong to any of the MDCT bases in the overcomplete dictionary, these correspond to the atoms that have been selected in the first iterations of the modified matching pursuit algorithm. These first bitplanes are encoded simply with the same standard bitplane encoder than in the previous approach. Then, Fig. 6.2 shows that only coefficients from the orthogonal dictionary remain after the fifth bitplane. These correspond to the atoms that have been selected after the switch in the modified matching pursuit algorithm. Consequently, all the coefficients that do not belong to the orthogonal dictionary are zeros after the fifth bitplane and the cost of encoding the significance map can be thus significantly reduced, by encoding the bits corresponding to the orthogonal dictionary only. In order that the decoder knows when the significance map size is reduced, it is necessary to add one bit per bitplane ("0" = full overcomplete dictionary, "1" orthogonal transform).

The algorithm is detailed in Alg. 14. The coefficients v_i are first normalized by the amplitude of the coefficient with maximum amplitude $A = max(abs(v_i))$ and the value A is quantized and transmitted (as in the standard bitplane encoder). Then, at each iteration, a test is performed. We decide to switch if all the non already significant coefficients that do not belong to the orthogonal dictionary are zeros. If it is the case, a bit "1" is emitted and the significance map is reduced to the coefficients of the orthogonal dictionary. Otherwise, a bit "0" is emitted and the whole significance map is encoded. The significance map is encoded using the same adaptive Golomb encoder used in the previous approach. The refinement pass is also the same.

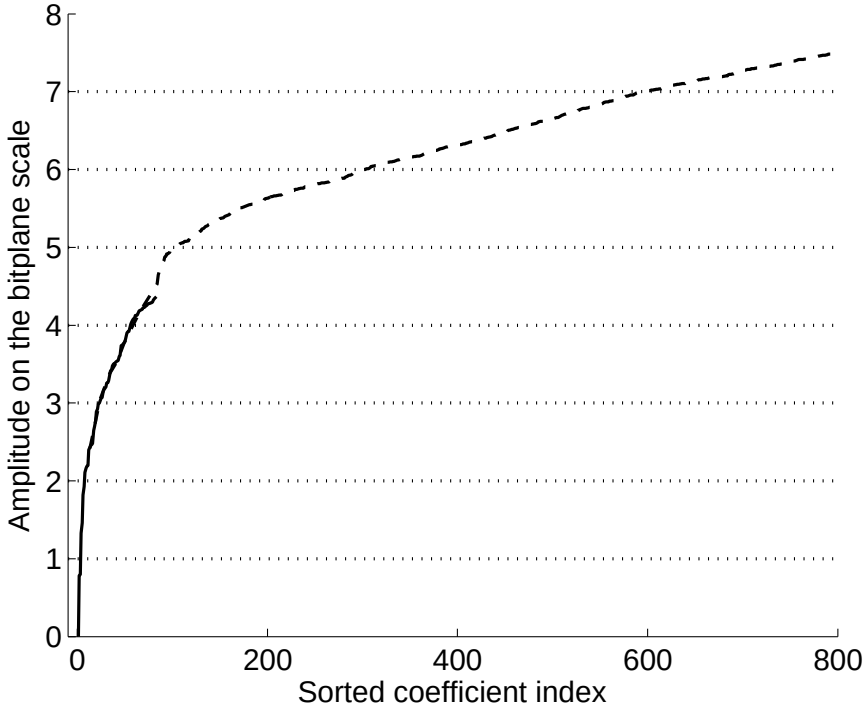

Figure 6.2: *Sorted coefficient amplitude on the bitplane scale. Dashed line corresponds to the coefficients that belong to the orthogonal dictionary and the plain line the other coefficients. Dotted lines correspond to the bitplane boundaries.*

Algorithm 14: Adaptive bitplane coding

Input: A vector of interleaved coefficients $\mathbf{v} = \{v_i | i = 1...ML_{M-1}\}$
Output: The bitstream

Quantize and code max amplitude $A = max(abs(v_i))$;
Initialization: $\mathbf{z} = \mathbf{0}$, $\mathbf{s} = \mathbf{1}$, $\nu = 1$;
repeat
  Test switch: switch if $v_i = 0$ for all i s.t. $z_i = 0$ and i not in orth. dict.;
  if *switch* **then**
    emit the bit "1";
    $s_i = 0$ for all i not in orth. dict.;
  else
    emit the bit "0";
  end
  Compute bitplane: $b_i = mod(floor(abs(v_i) * 2^{\nu}/A), 2)$ for all i s.t. $s_i = 1$;
  Significance pass: code $BS = \{b_i | z_i = 0$ and $s_i = 1\}$ and signs;
  Refinement pass: code $BR = \{b_i | z_i = 1\}$;
  Update significance: $z_i = 1$ for all i s.t. $b_i \in BS$ and $b_i = 1$;
  Iterate: $\nu = \nu + 1$;
until *bit budget spent or* $\nu > \nu_{max}$;

6.3.2 Optimal switching parameter

The proposed adaptive MP depends only on one parameter SW, which is the energy decay per coefficient under which it is better to switch from overcomplete to orthogonal. Of course, SW can be estimated on-line, by computing at each iteration the most favorable rate-distortion configuration. However, since the coefficients are not encoded separately but in bitplanes, this technique leads to a significant complexity increase (at each iteration, one has to "look ahead" a large number of steps to decide on the best strategy). Instead, we would like to compute a fixed value for SW that is approximately optimal (in terms of rate-distortion), without any additional computation in the MP loop. It should be noted that we tried other methods (e.g. based on SNR, based on the coefficients amplitude) but they were too signal-dependant.

In the following, we consider one particular slot $\mathcal{S}_q$, and we suppose that the switch is done for that timeslot at the iteration $n_q(i_{switch})$. We make the assumption that the SNR decay DEC_q in that timeslot is constant for the few iterations following the switch (this is approximately verified if the switch does not occur in the first iterations). Then, we consider two cases: if at iteration $n_q(i_{switch})$, we stay in the overcomplete dictionary (no switch), the decay value would be DEC_q^O; if at iteration $n_q(i_{switch})$, the dictionary is reduced to an orthogonal dictionary (switch), the decay value would be lower and equal to DEC_q^T. We now make the empirical observation that there exists a simple relation between DEC_q^O and DEC_q^T. We decompose a 20 seconds signal composed of several audio types contents (monophonic, polyphonic) with the adaptive MP and several values of the parameter SW. We also decompose the same signal in the overcomplete dictionary with the standard MP (no switch) as a reference to compute DEC_q^O. In each timeslot, and for each parameter value, we compute the mean of the decay on the 100 iterations $n_q(i)$ following the switch at iteration $n_q(i_{switch})$ for each case and we plot DEC_q^T as a function of DEC_q^O. Fig. 6.3 shows the obtained values and a linear approximation. We thus approximate the relation between DEC_q^T and DEC_q^O as: $DEC_t = \gamma DEC_o$ with $\gamma = 0.6$.

Now, as we have supposed that the SNR decay DEC_q in a timeslot is constant for the few iterations following the switch, and if we neglect the influence of the neighbouring timeslots $\mathcal{S}_{q-1}$ and $\mathcal{S}_{q+1}$, the energy and the absolute value of the coefficients have approximately the same decay: DEC_q^O if we remain in the overcomplete dictionary, and DEC_q^T if we switch to the orthogonal dictionary. As a bitplane corresponds to a division per two of the absolute value of the coefficients, we have

$$10^{-Nb^T DEC_q^T/20} = \frac{1}{2} \tag{6.9}$$

with Nb^T the approximate number of coefficients per bitplane, which is then equal to

$$Nb^T = \frac{20}{DEC_q^T \log_2 10} \tag{6.10}$$

for the orthogonal dictionary and

$$Nb^O = \frac{20}{DEC_q^O \log_2 10} \tag{6.11}$$

for the overcomplete dictionary. As the major contribution to the bitrate is due to coding the significance maps, in following computations we neglect the sign and refinements bits. Assuming that the coding cost for the significance map can be estimated by entropy, we can then compute the average rate per significant coefficient as

$$R^T = \frac{1}{p}\left(-p \log_2(p) - (1-p) \log_2(1-p)\right) \tag{6.12}$$

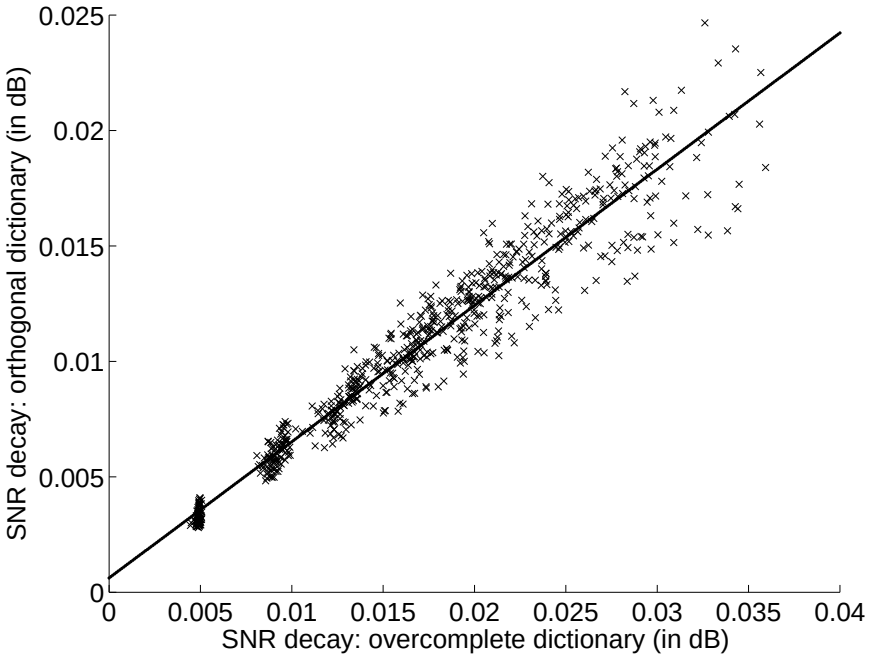

Figure 6.3: *Decay of the orthogonal dictionary as a function of the overcomplete dictionary; the decay is computed on the* 100 *iterations following the switch.*

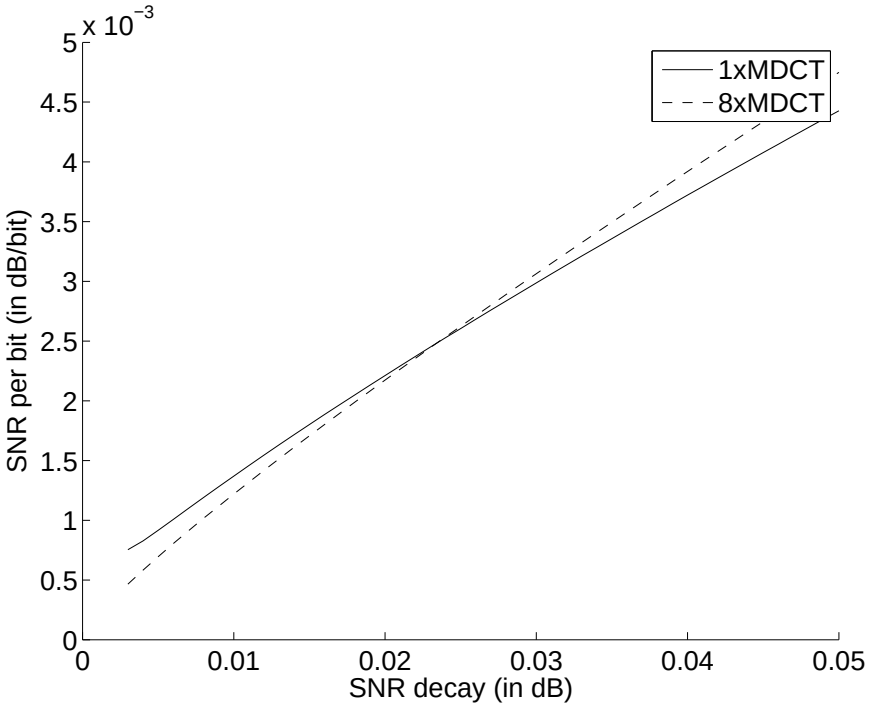

Figure 6.4: *Distortion per bits as a function of the SNR decay for two dictionaries.*

with $p = \frac{Nb^T}{S_{len}}$ for the orthogonal dictionary and

$$R^O = \frac{1}{\tilde{p}} \left(-\tilde{p} \log_2(\tilde{p}) - (1-\tilde{p}) \log_2(1-\tilde{p}) \right) \tag{6.13}$$

with $\tilde{p} = \frac{Nb^O}{MS_{len}}$ for the overcomplete dictionary. In short, each coefficient in the bitplane after the switch iteration decreases the SNR by DEC_q^O dB at a cost of R^O bits if we remain the overcomplete dictionary, and decreases the SNR by DEC_q^T dB at a cost of R^T bits if we switch to the orthogonal dictionary. Consequently, we decide to switch to the orthogonal dictionary if $DEC_q^T/R^T > DEC_q^O/R^O$. We are now able to compute the optimal value SW numerically. Fig. 6.4 plots DEC_q^T/R^T and DEC_q^O/R^O as a function of DEC_q^O for $\gamma = 0.6$. We finally find numerically the optimal parameter value which is approximately $SW = 0.025$ (numerical simulations have shown that the exact value is not really critical).

6.4 Evaluation

6.4.1 Performance

We have tested our algorithm on the same signals as used in the previous chapter: bagp, gloc, harp, orch, popm. We compare three coders based on three different signal representations. First, the two reference coders are a transform coder based on a single MDCT ($M = 1$) with an analysis window length of 2048 samples, and the overcomplete approach coder with the standard MP in a union of 8 MDCT bases. These two coders are compared with the novel coder proposed in this chapter, based on the modified MP with an adaptive dictionary that switches from 8xMDCT to 1xMDCT (with length 2048 samples). We remark that contrary to the previous chapter, the evaluation measure is based here on SNR as the decomposition algorithm is based on SNR too. More relevant objective measure for audio coding such as PEMO-Q [HK06] or even listening tests are planned for future work. The results are shown on Fig. 6.5. It clearly shows that the performance of the new coder is the same as the previous approach at low bitrates and the same as transform coding at high bitrates, even slightly higher.

6.4.2 Computation times

Fig. 6.1 compare the computation time needed to code the six files with the three approaches: the single MDCT coder, the proposed adaptive MP coder and the standard MP coder (with a precision of 60 dB). Though it is still about 100 times slower than a single MDCT, the new approach is much faster than the previous approach.

	MDCT	Adaptive MP	MP (60dB)
bagp	0.03	3.88	140.60
gloc	0.03	2.11	42.41
harp	0.03	2.00	146.29
orch	0.03	3.89	151.90
popm	0.03	2.05	214.06

Table 6.1: *Normalized computation times on a Core 2 Duo 2.0 GHz laptop (in seconds/seconds)*

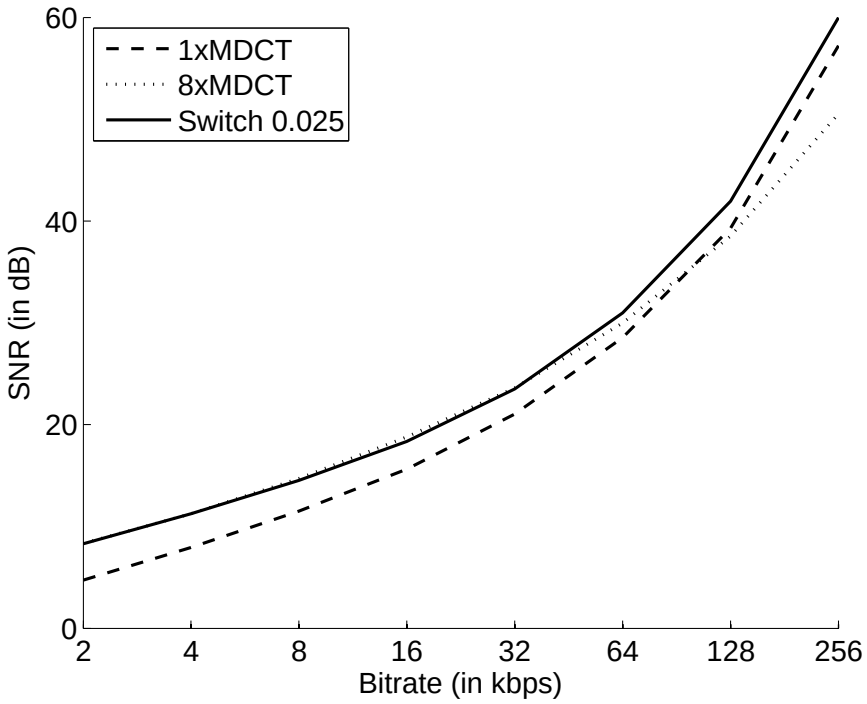

Figure 6.5: *Mean SNR for six signals in function of the bitrate for 3 coders: transform coder with 1xMDCT, standard MP with 8xMDCT, adaptive MP from 8xMDCT to 1xMDCT with a switch parameter value of SW = 0.025.*

6.5 Conclusion

The scalable audio coder proposed in the previous chapter gave better performance than transform-based coding at low bitrates but slightly worse performance at high bitrates. The signal representation was based on a standard Matching Pursuit decomposition in an overcomplete union of MDCT bases. We have shown that the energy decay in this overcomplete dictionary is high on the first iterations and becomes almost equal to the decay of an orthogonal dictionary after some iterations. As it is less costly to encode atoms in an orthogonal dictionary, it is better from a coding point of view to decompose the residual in an orthogonal dictionary when the energy decay becomes too low. We thus have proposed a modified MP which switches from an overcomplete dictionary to an orthogonal dictionary when the energy decay is below a threshold. We have then derived an optimal switching parameter value. Finally, we have shown experimentally that the resulting coder reaches the performance of the previous approach at low bitrates and the performance of a transform coder at high bitrates.

This study also raises some questions: first, it is not clear whether there is a fundamental reason for such a simple (approximate) relationship between DEC_q^O and DEC_q^T. Second, this leads to wonder if there are more signal-independent techniques to perform the switch near the optimum. Finally, further studies will have to investigate whether the rate-distortion optimization as performed here, with distortion as mean quadratic error, is also optimal from a perceptual point of view.

Chapter 7

Embedded Polar Quantization

Abstract

This chapter proposes simple algorithms for embedded polar quantization. Sets of constrained-resolution embedded quantizers are built recursively by successive refinement processes, that are detailed for strict polar quantization and unrestricted polar quantization. The quadratic error minimization problem is solved using equations similar to those of Max, and the refinement algorithm can, in the unrestricted case, be simplified using a high-rate approximation. For Gaussian data, comparisons with reference non-embedded quantizers show that the embedding property comes at an often negligible cost in terms of rate-distortion performance. The results presented in this chapter have been published in IEEE Signal Processing Letters [RD07].

Contents

7.1 Introduction

We have studied in the previous chapters one overcomplete dictionary based on MDCT atoms, which is the union of several MDCT with different scales. One can imagine other ways to build overcomplete dictionaries. For example, one possibility is to use a complex extension of the MDCT, which is also called Modulated Complex Lapped Transform (MCLT) [Mal99b]. Contrary to the MDCT, the MCLT has the interesting property of phase-invariance, which is useful for coding time structures. MCLT-based coding has already been investigated in [DD06] and [YM08]. For a N-length signal, the MCLT dictionary is composed by N complex atoms and it is thus a twice overcomplete dictionary. The corresponding signal model is given by

$$\mathbf{x} = \Re\left(\mathbf{\Phi c}\right) + \mathbf{r} \tag{7.1}$$

with $\mathbf{x}$ the signal, $\mathbf{r}$ the residual, $\mathbf{c}$ the vector of (complex) coefficients of length N and $\mathbf{\Phi}$ the synthesis matrix of size $N \times N$ defined as

$$\mathbf{\Phi}\left(n, pK + k\right) = g_{p,k}(n) \ , \ 0 \leq p < P \ , \ 0 \leq k < K \ , \ 0 \leq n < N \tag{7.2}$$

where P is such that $N = PK$ and is the number of segments of length $K = L/2$, and

$$g_{p,k}(n) = w_p(u)\sqrt{\frac{2}{K}} \exp\left[i\frac{\pi}{K}\left(u + \frac{K}{2} + \frac{1}{2}\right)\left(k + \frac{1}{2}\right)\right] \tag{7.3}$$

and

$$u = n - \left(p - \frac{1}{2}\right)K \tag{7.4}$$

and the window $w_p(u)$ is a sine-based window. A signal is approximated in such a dictionary using a two-dimensional matching pursuit based on the projection of the signal in the subspace composed by the complex atoms and their conjugates as described in [Goo97]. The algorithm has been implemented in MPTK. In the following, we describe a small experiment that shows the potential benefit of using a MCLT instead of a single MDCT in an audio codec. The five signals of Annex A are approximated in a MCLT dictionary with window size 2048 samples. The vector of complex coefficients is then represented in magnitude/phase form and quantized using a polar quantizer. The SNR as a function of the estimated bitrate is plotted on Fig. 7.1. This shows that gain may be obtained at very low bitrates. However, to use efficient bitplane encoding techniques similar to those presented in the last chapter, it is necessary to design embedded quantizers for 2-dimension circularly symmetric data. We thus propose several embedded polar quantizers for such data. This preliminary study is just the first step towards a fully scalable audio coder based on MCLT. As we did not have time to go further in that direction, we will only present in this chapter the proposed embedded polar quantizers.

Constrained-resolution embedded quantization as stated in this chapter can be defined as follows. Consider K quantizers $\mathcal{Q}^{(k)}, k = 1, ..., K$

$$\mathcal{Q}^{(k)} : \mathbf{R}^L \rightarrow \left\{0, .., 2^k - 1\right\}, \tag{7.5}$$

with L the dimension of the sample space and $N^{(k)} = 2^k$, the number of cells for quantizer $\mathcal{Q}^{(k)}$.

Embedded quantization was first introduced for scalar quantization ($L = 1$) in [Tzo86]. An optimal quantizer is chosen first as a reference, it is designed using the standard numerical optimization methods of Max [Max60]. The quantizers of lower and higher rates are designed by

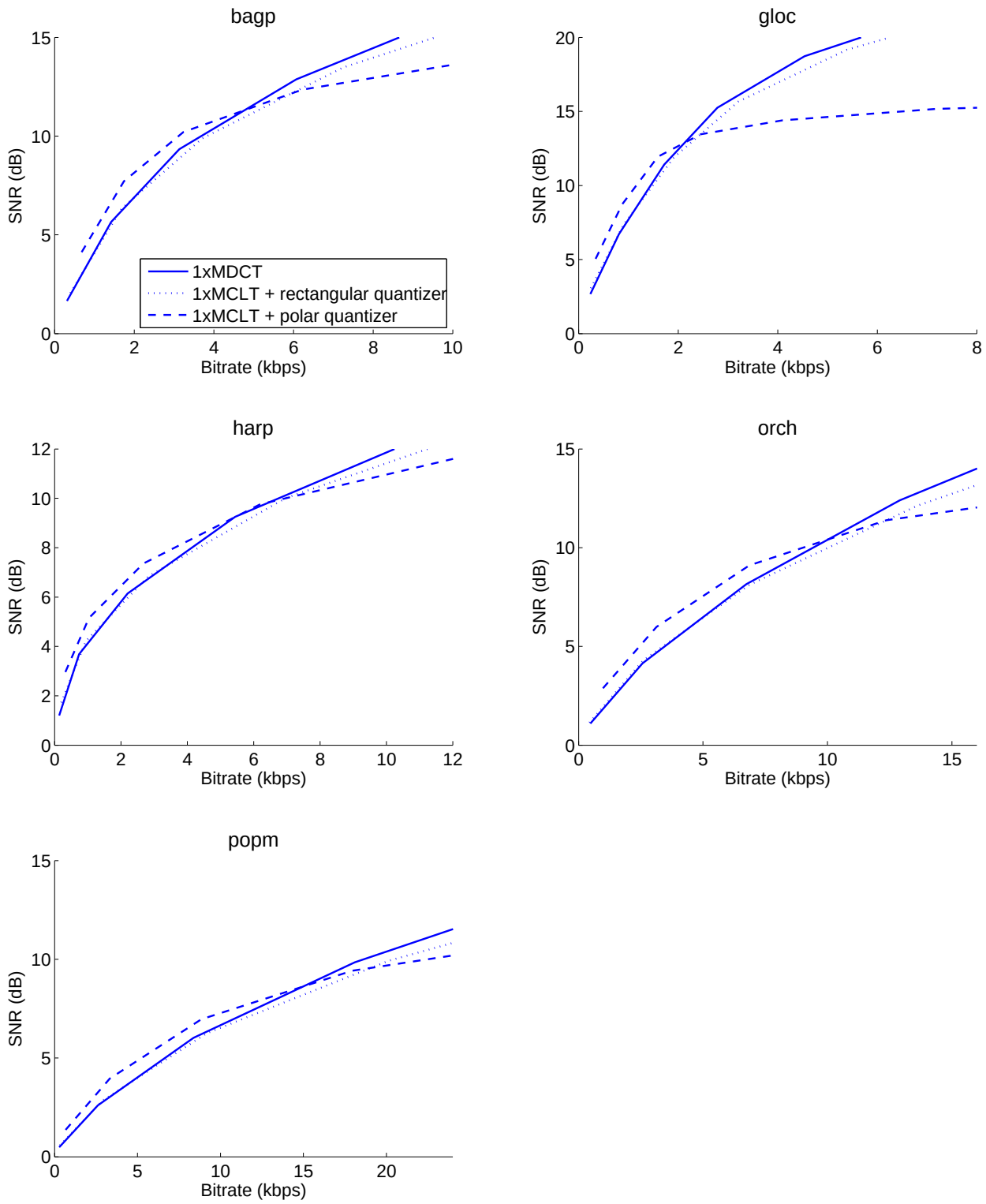

Figure 7.1: SNR as a function of the estimated bitrate using simple 2-dimension quantizers.

iteratively aligning the quantization thresholds and optimizing the thresholds and reconstruction points using methods similar to [Max60]. Such quantizers can be easily represented using binary trees. Each stage of the tree corresponds to a particular rate, the nodes correspond to the boundary points, and the branches correspond to the output levels. Practical embedded scalar quantizers have also been proposed in [KLK02, BF96, BF98]. Finally, embedded quantization has been generalized to vector quantization. An example of such vector quantizers are the Tree-Structured Vector Quantizers (TSVQ) [GG92]. The 2-ary TSVQ is designed using a binary tree and numerical solving methods similar to [Max60], the embedded quantizers are then the pruned tree corresponding to the first k stages of the TSVQ.

We consider a particular case of 2-dimensional ($L = 2$) quantizers which enable embedded polar quantization. Polar quantizers are natural quantizers for 2-dimensional data with circularly symmetric densities. Amplitude and phase can be quantized separately, this case is called Strict Polar Quantization (SPQ, [Pea79]) or they can be quantized jointly, this is Unrestricted Polar Quantization (UPQ, [Wil80]).

In this chapter, a circularly symmetric complex variable x is used, represented by its polar coordinates (r, θ). Since amplitude and phase variables are independent, the variable joint density function is then

$$f_X(r, \theta) = \frac{1}{2\pi} f_R(r), \tag{7.6}$$

with $f_R(r)$ is the marginal density function of the amplitude variable. A polar quantizer $\mathcal{Q}^{(k)}$ with $N^{(k)}$ cells is defined as follows. The amplitude range is partitioned into $M^{(k)}$ levels and within each amplitude level, indexed by $m = 1, 2, ..., M^{(k)}$, there are $P_m^{(k)}$ equal-sized phase cells, such that $\sum_{m=1}^{M^{(k)}} P_m^{(k)} = N^{(k)}$. Note that for SPQ, one has $P_m^{(k)} = P^{(k)}$ for all m. The boundaries of the amplitude levels are

$$R_0^{(k)} = 0 < R_1^{(k)} < R_2^{(k)} < ... < R_M^{(k)} = \infty, \tag{7.7}$$

and the boundaries of the phase regions for the amplitude level m are

$$0 < \frac{2\pi}{P_m^{(k)}} < 2\frac{2\pi}{P_m^{(k)}} < ... < (P_m^{(k)} - 1)\frac{2\pi}{P_m^{(k)}} < 2\pi. \tag{7.8}$$

The output point for the cell $\mathcal{R}_{m,p}^{(k)}$ defined by the amplitude level m and the phase region p is $(\alpha_m^{(k)}, \beta_{m,p}^{(k)})$. The mean square error is then expressed by

$$D^{(k)} = \sum_{m=1}^{M^{(k)}} \sum_{p=1}^{P_m^{(k)}} \iint_{\mathcal{R}_{m,p}^{(k)}} \left| re^{j\theta} - \alpha_m^{(k)} e^{j\beta_{m,p}^{(k)}} \right|^2 f_X(r, \theta) dr d\theta. \tag{7.9}$$

The purpose of this chapter is to detail simple construction rules for embedded quantizers, in the cases of SPQ (section 7.2) and UPQ (section 7.3). We compare the performance of the embedded quantizers to the performance of the non-embedded quantizers using Gaussian data.

7.2 Embedded Strict Polar Quantization

7.2.1 Quantizers design

Strict polar quantization (SPQ) is first considered. As amplitude and phase are quantized separately, scalar embedded quantization techniques can easily be used to design the embedded SPQ. An algorithm similar to [Tzo86] is used, where an optimal reference quantizer is first selected.

Then, successive lower- and higher-rate quantizers are built in a recursive manner. The design algorithm consists of three steps:

1. Reference quantizer: An optimal SPQ $\mathcal{Q}^{(k_{ref})}$ is designed using the algorithm of [Pea79]. $\mathcal{Q}^{(k_{ref})}$ has a number of amplitude levels $M^{(k_{ref})}$ and a number of phase levels $P^{(k_{ref})}$.

2. Lower rate quantizers: Two lower-rate quantizers $\mathcal{Q}^{(k)}$ are designed from $\mathcal{Q}^{(k+1)}$. The first one is designed from $\mathcal{Q}^{(k+1)}$ by selecting only every second amplitude boundary and keeping the same number of phase levels. The other one is designed from $\mathcal{Q}^{(k+1)}$ by selecting only every second phase boundary and keeping the same number of amplitude levels. The output points for the two quantizers are found using

$$\alpha_m^{(k)} = S(P^{(k)}) \frac{\int_{R_{m-1}^{(k)}}^{R_m^{(k)}} r f_R(r) dr}{\int_{R_{m-1}^{(k)}}^{R_m^{(k)}} f_R(r) dr} \tag{7.10}$$

$$\beta_{m,p}^{(k)} = \left(p - \frac{1}{2}\right) \frac{2\pi}{P^{(k)}}, \tag{7.11}$$

with $S\left(P^{(k)}\right) = sinc\left(\pi/P^{(k)}\right)$. Then, the distortion is computed for each quantizer using Eq. (7.9) and the one with the minimum distortion is selected.

3. Higher rate quantizers: Similarly, two higher-rate quantizers $\mathcal{Q}^{(k)}$ are designed from $\mathcal{Q}^{(k-1)}$. The first one is designed from $\mathcal{Q}^{(k-1)}$ by refining the amplitude quantizer by two and keeping the same number of phase levels. The new amplitude boundaries are initialized at the middle of the previous amplitude boundaries. Boundaries and output points are then computed iteratively by first updating the output points using equations (7.10) and (7.11) and next by updating the new amplitude boundaries using

$$R_m^{(k)} = \frac{1}{S(P^{(k)})} \frac{\alpha_m^{(k)} + \alpha_{m+1}^{(k)}}{2}. \tag{7.12}$$

Another higher-rate quantizer can be designed from $\mathcal{Q}^{(k-1)}$ by refining the phase quantizer by two and keeping the same number of amplitude levels. The new phase boundaries are at the middle of the previous boundaries and the new output points are calculated using Eq. (7.10) and (7.11). Again, the distortion is computed for each quantizer using Eq. (7.9) and the one with the minimum distortion is selected.

7.2.2 Application: Gaussian data

The performance of our quantizers are evaluated using an independent bivariate Gaussian source. Such a source gives a good base of comparison as the results for existing quantizers are well-known [Pea79, Wil80]. Moreover, the source produced by the DFT of a stationary signal of length N tends to a bivariate Gaussian source as N becomes large. A bivariate Gaussian source is equivalent to an independent circularly symmetric complex variable whose amplitude has a Rayleigh distribution. The joint density function is then $f_X(r,\theta) = \frac{1}{2\pi} f_R(r)$ with $f_R(r) = \frac{r}{\sigma^2} \exp\left(-\frac{r^2}{2\sigma^2}\right)$. Several embedded SPQs are compared using different values for k_{ref} and these embedded SPQs are also compared with the (non-embedded) optimal SPQ (from [Pea79]).

Fig. 7.2 shows the results obtained for $k_{ref} = 1, 6, 8, 12$ bits. When the actual number of bits is not far from k_{ref}, the excess distortion of the embedded quantizers is low, typically less than 0.2 dB. The choice of k_{ref} will depend on the application: a low k_{ref} is selected if the probability

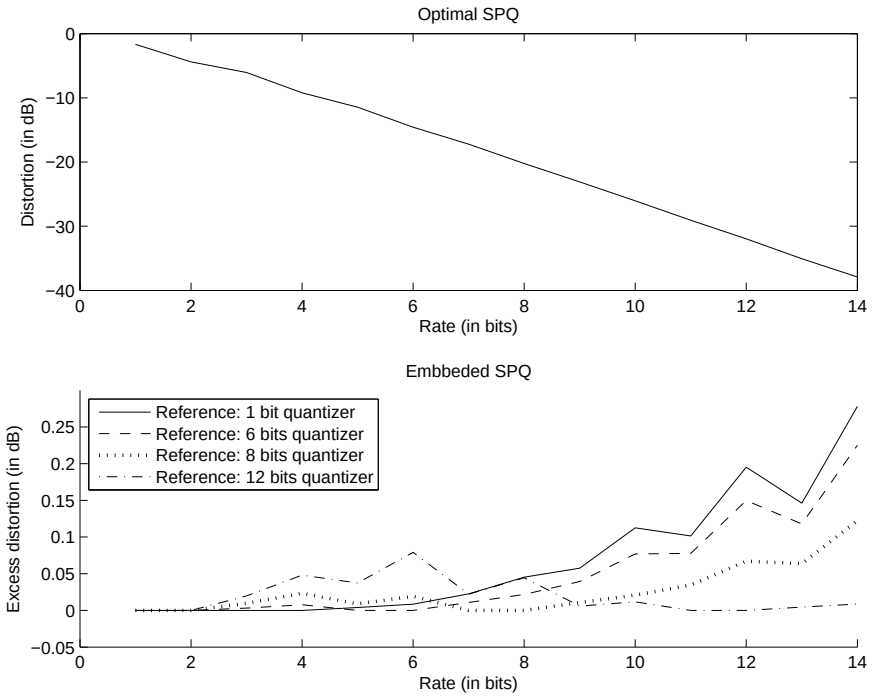

Figure 7.2: **(top)** Rate-Distortion curve for the optimal SPQ of [Pea79]. **(bottom)** Excess distortion over the optimal SPQ for several embedded SPQs using different values for k_{ref}.

of having low rates is high; and a high k_{ref} is selected if the probability of having low rates is low. Note that in cases where little is known *a priori* about such probability, a conservative approach should favor a large k_{ref}.

7.3 Embedded Unrestricted Polar Quantization

7.3.1 Quantizers design

Consider the more general case of unrestricted polar quantization (UPQ). Here, lower rates embedded quantizers cannot necessarily be found using an optimal reference quantizer, and consequently the same design method as in SPQ cannot be used. Indeed, an optimal UPQ as designed in [Wil80] does not necessarily have a power-of-two number of phase regions at a given amplitude level. Instead, an algorithm similar to 2-ary TSVQ is used. The design algorithm consists of two steps:

1. Reference quantizer: The first quantizer $\mathcal{Q}^{(1)}$ is the optimal 2-cell quantizer (top-left of Fig. 7.5).

2. Higher rate quantizers: For a higher rate quantizer $\mathcal{Q}^{(k)}$ with $k = 2, ..., K$, each region of $\mathcal{Q}^{(k-1)}$ are refined separately in two different ways, either by refining the amplitude or the phase. This leads to $2^{2^{k-1}}$ possibilities for $\mathcal{Q}^{(k)}$. However, as the source is supposed to be circularly symmetric, cells at the same amplitude level perform in the same way. Consequently, there are only $2^{M^{(k-1)}}$ possibilities for $\mathcal{Q}^{(k)}$. For a cell at a given amplitude level, one has to decide whether an amplitude refinement or a phase refinement gives the smallest distortion; this choice is discussed in the next subsection. Then, the new boundaries and output points are computed using equations

similar to those of Max [Max60]. The same process is repeated for each amplitude level.

7.3.2 Choice between amplitude and phase refinement

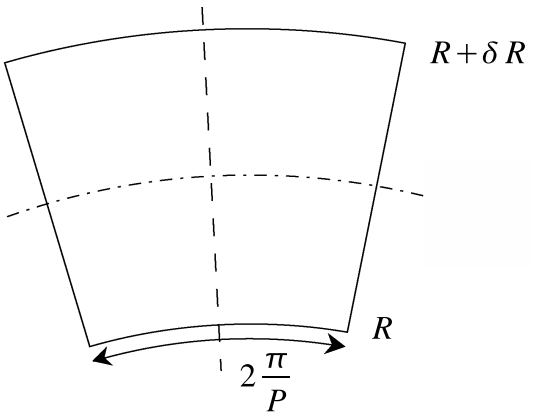

Figure 7.3: A cell delimited by the amplitude boundaries R and $R+\delta R$, and the phase boundaries difference $\frac{2\pi}{P}$. The dashed line represents the new cells boundary for phase refinement, and the dash-dotted line the new cells boundary for amplitude refinement.

Consider a cell delimited by the amplitude boundaries R and $R+\delta R$, and the phase boundaries difference $\frac{2\pi}{P}$ (see Fig. 7.3). Depending on these three parameters, it may be better to choose either an amplitude refinement (the new divided cell is noted (a)) or a phase refinement (the new divided cell is noted (b)). For amplitude refinement, the amplitude boundary $R+\gamma$ and the two new output points α_1 and α_2 are calculated iteratively using the following equations:

$$R+\gamma = \frac{1}{S(P)}\frac{\alpha_1+\alpha_2}{2} \tag{7.13}$$

$$\alpha_1 = S(P)\frac{\int_R^{R+\gamma} r f_R(r)dr}{\int_R^{R+\gamma} f_R(r)dr} \tag{7.14}$$

$$\alpha_2 = S(P)\frac{\int_{R+\gamma}^{R+\delta R} r f_R(r)dr}{\int_{R+\gamma}^{R+\delta R} f_R(r)dr}. \tag{7.15}$$

For the phase refinement, the phase boundary is at the middle of the cell and the new output point α is calculated using

$$\alpha = S(2P)\frac{\int_R^{R+\delta R} r f_R(r)dr}{\int_R^{R+\delta R} f_R(r)dr}. \tag{7.16}$$

The distortions D_a and D_b are computed using Eq. (7.9), and the one that gives the smallest distortion is selected.

An analytical solution for the refinement choice can be derived using high-rate approximation. The joint probability density function is supposed to be constant in the cell. Moreover, P is supposed to be large and $S(P)$ is approximated using its second-order Taylor expansion. The

total distortion D_a for the cell (a) is

$$\begin{aligned} D_a.\frac{P}{f_R(R)} &= \frac{(R+\delta R-\alpha_2)^3-(R+\delta R/2-\alpha_2)^3}{3} \\ &+\frac{\pi^2}{3P^2}((R+\delta R)^2-(R+\delta R/2)^2)\frac{\alpha_2}{2} \\ &+\frac{(R+\delta R/2-\alpha_1)^3-(R-\alpha_1)^3}{3} \\ &+\frac{\pi^2}{3P^2}((R+\delta R/2)^2-R^2)\frac{\alpha_1}{2}, \end{aligned}$$

with

$$\begin{aligned} \alpha_1 &= \left(1-\frac{\pi^2}{6P^2}\right)\left(R+\frac{\delta R}{4}\right) \\ \alpha_2 &= \left(1-\frac{\pi^2}{6P^2}\right)\left(R+3\frac{\delta R}{4}\right). \end{aligned}$$

The total distortion D_b for the cell (b) is

$$\begin{aligned} D_b.\frac{P}{f_R(R)} &= \frac{(R+\delta R-\alpha)^3-(R-\alpha)^3}{3} \\ &+\frac{\pi^2}{3(2P)^2}((R+\delta R)^2-R^2)\frac{\alpha}{2}, \end{aligned}$$

with

$$\alpha=\left(1-\frac{\pi^2}{6(2P)^2}\right)\left(R+\frac{\delta R}{2}\right). \tag{7.17}$$

To find the boundaries, we need to solve $D_a \leq D_b$. Using a first-order approximation of the former expressions, this inequality is equivalent to:

$$\frac{P}{2\pi}\geq\left(\frac{R}{\delta R}+\frac{1}{2}\right). \tag{7.18}$$

The inequality (7.18) has a simple interpretation: if the length δR of the radial boundary of the cell is bigger than $(R+\delta R/2)\frac{2\pi}{P}$ which is the length of the average circle arc, then amplitude refinement is preferred, and vice-versa. In the next subsection, this analytical solution appears to be a good approximation, even at low rates. It saves computational cost since only boundaries and output points need to be calculated.

7.3.3 Application: Gaussian data

The performance of our quantizer is evaluated using the same independent bivariate Gaussian source as in section 7.2.2. The following quantizers are compared: the optimal UPQ [Wil80], the high-rate optimal UPQ [SK86], and our embedded UPQ (without the high-rate approximation for the refinement choice). For the optimal UPQ, the simulation has been performed up to a rate of 6 bits only since for higher rates the computational cost is too high.

To evaluate the excess distortion due to the high-rate approximation for the refinement choice, the excess distortion over the embedded UPQ is plot for the embedded UPQ with the high-rate approximation for the refinement choice (Fig. 7.4). The excess distortion for the embedding property here is always less than 1 dB over optimal non-embedded, a price to pay that can, in certain applications, be considered small with regard to its benefits. Finally, the first six embedded UPQ are shown in Fig. 7.5, using the high-rate approximation for the refinement choice.

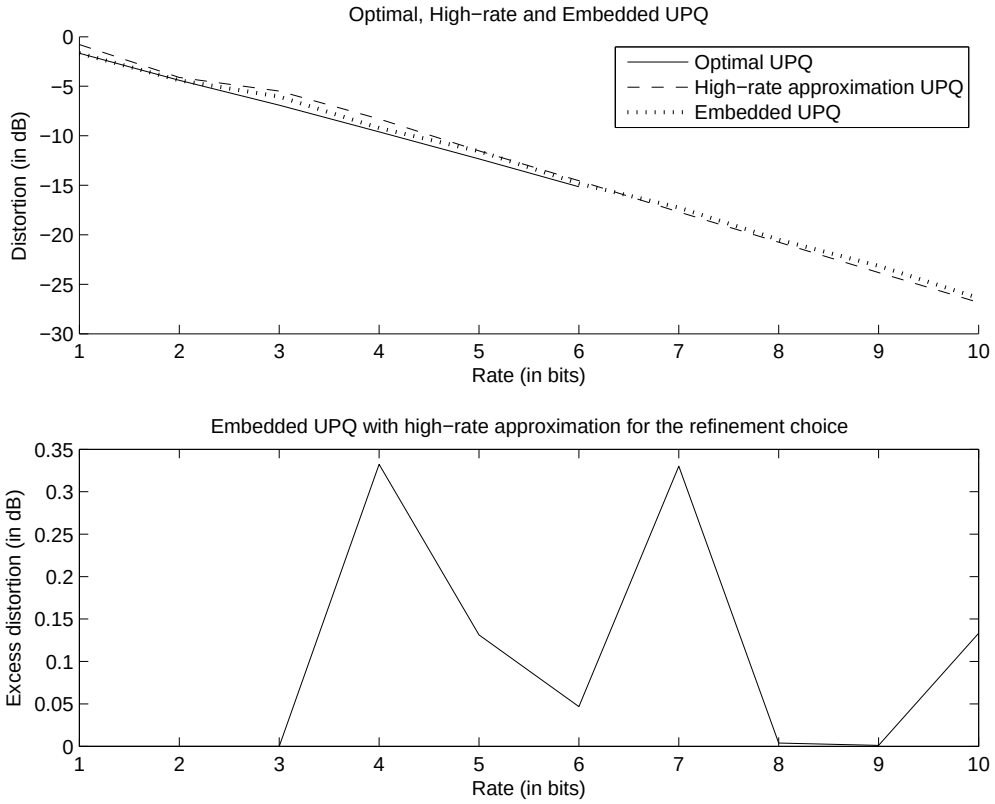

Figure 7.4: **(top)** Rate-Distortion curves for the optimal UPQ of [Wil80], the optimal high-rate UPQ of [SK86] and our embedded UPQ (without the high-rate approximation for the refinement choice). **(bottom)** Excess distortion over the embedded UPQ for the embedded UPQ with the high-rate approximation for the refinement choice

7.4 Conclusions

In this chapter, we have described simple algorithms to design polar quantizers that can be embedded. Two kinds of quantizers have been distinguished: embedded strict polar quantizers where amplitude and phase are quantized separately; and embedded unrestricted polar quantizers where amplitude and phase are quantized jointly. For bivariate Gaussian data, this results in a small increase of distortion over when compared to the standard optimal polar quantizers (typically less than 0.2 dB for SPQ, less than 1 dB for UPQ). A high-rate approximation for the refinement choice has been introduced and results in negligible excess distortion.

Future work would investigate the integration of the proposed quantizers in a bitplane encoder. This would require considering a zero central cell, possibly more than two cells for quantizing the initial phase, possibly more than one refinement bit at each step of the algorithm. The bitplane encoder would then be associated to a MCLT-based signal representation and the resulting audio codec would need a proper evaluation. It is also necessary to consider other amplitude distributions, such as the generalized Gaussian distribution which is closer to the real distribution of the MCLT coefficient amplitude, or even a simple uniform distribution (as in previous approach).

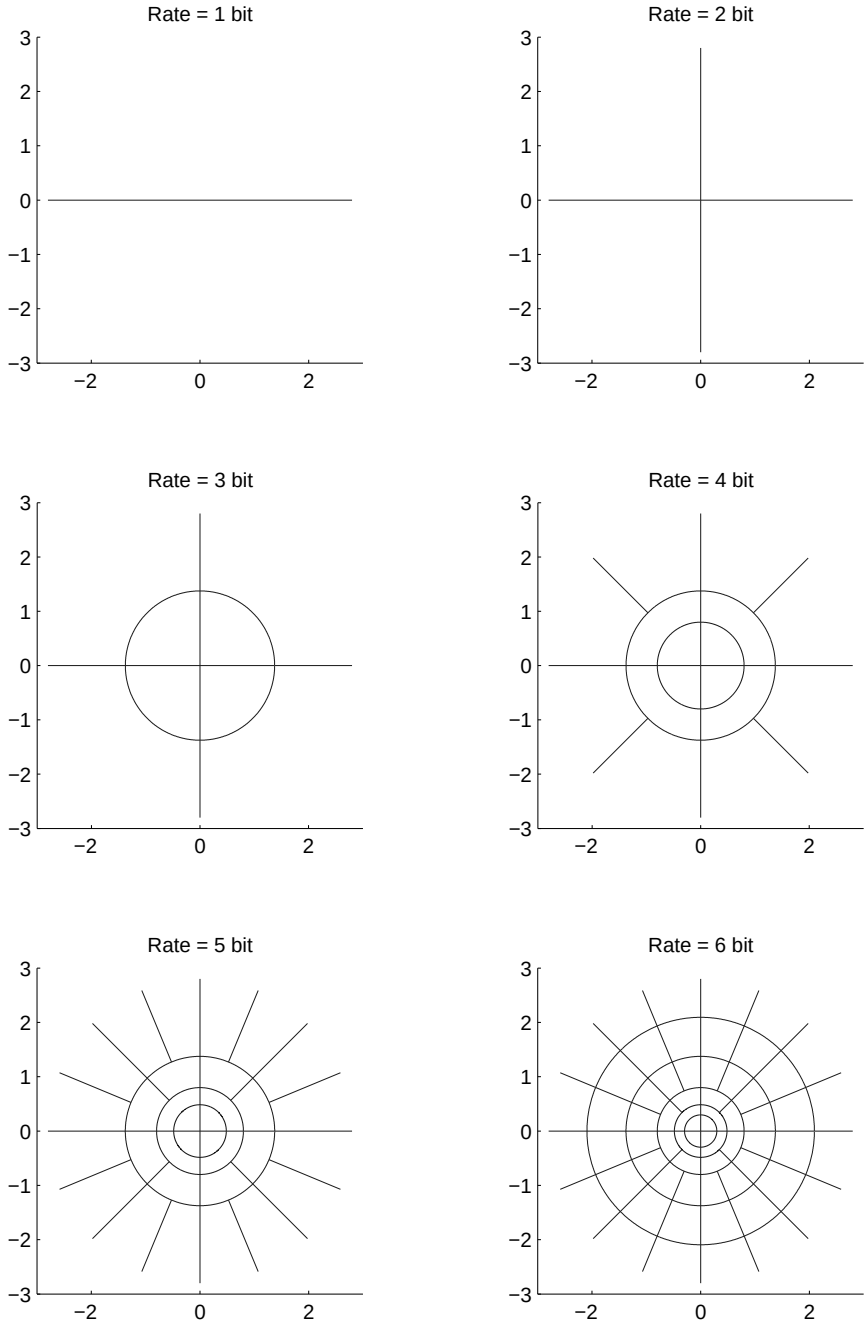

Figure 7.5: First six embedded UPQs for Gaussian data.

Chapter 8

Transform-domain Audio Indexing

Abstract

This chapter studies transform-domain audio indexing. We propose simple algorithms for the computation of several mid-level representations directly from transform coefficients. Three applications are considered , respectively beat tracking, chord recognition and musical genre classification. The performance for three coders are compared and discussed: two standard filter-bank based audio codecs (MP3 and AAC) and one proposed audio codec. A subset of the results presented in this chapter have been published in the proceedings of ISMIR 2008 [RRD08a].

Contents

8.1 Introduction

On the one hand, music recordings are widely available in coded format. The reason is that state-of-the-art audio coders such as MP3 [ISO92] or AAC [ISO01] are able to reduce the size of a PCM audio signal more than 10 times, while guaranteeing a near-transparent quality. Consequently, such technology allows users to easily exchange and store music on mobile devices and networks.

On the other hand, state-of-the-art audio indexing algorithms such as beat tracking [KEA06, DP07] and chord recognition [BP05, LS08] are designed to process PCM audio signals. Consequently, to use them on coded audio, one has to decode to PCM first and then apply the audio indexing algorithm on the PCM signal (*Processing in the time domain*, see Fig. 8.1). To save computational cost, which is often required for example when using such algorithms with mobile devices or on very large databases, it would be more efficient to design audio indexing algorithms that work directly with the coded data.

There are two ways to process coded data depending on which stage of the decoding process we are working on (see Fig. 8.1). The first way is to use directly the bitstream, this approach is called *processing in the compressed domain*. The second way is to use the transform representation, this approach is called *processing in the transform domain*. The first approach is faster as we avoid the cost of decoding the transform representation, however, for certain cases the information available in the bitstream is not sufficiently explicit, and it it thus necessary to use the transform domain representation. In this chapter, we only study transform domain audio indexing.

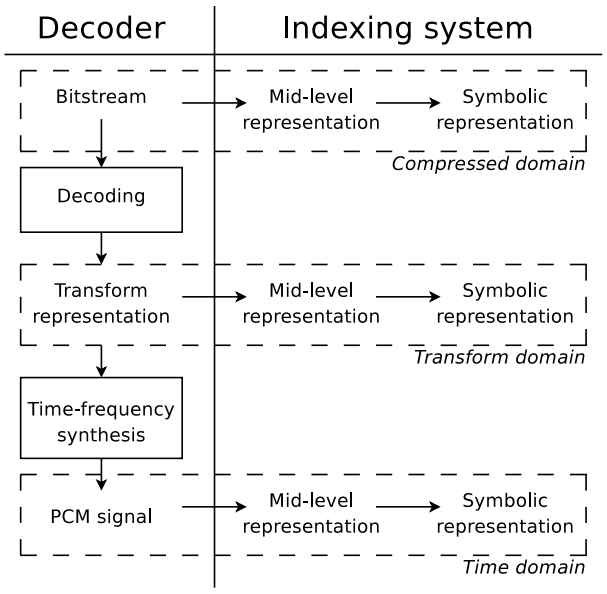

Figure 8.1: *Block diagram of a common audio decoder and three possible audio indexing systems.*

Most existing work on transform-domain audio indexing deal with MPEG-1 coded data (a good review on MPEG-1 transform-domain audio indexing is the technical report of Pfeiffer and Vincent [PV01]). Patel and Sethi [PS96] was probably the first study to propose low-level audio features based on MPEG-1 compressed data. The basic principle is to use the internal MPEG-1 representation composed by the 32 PQF subband signals, and to compute low-level features such as "signal energy", "pause rate", "band energy ratio"... These audio features were then combined

with video features and used in a machine learning system in order to classify video clips. Other works [YZ97, NLS+99, TC00, WV01, SXWK04, ZW08] follow a similar approach but propose different low-level audio features and consider different applications such as speech recognition [YZ97], audio segmentation and classification [NLS+99, TC00], beat tracking [WV01, ZW08] and music summarization [SXWK04]. One should note that the work of Wang et al. [WV01, SXWK04, ZW08] is in a way different from other works as they use MDCT coefficients for the calculation of audio features. Their approach is thus limited to MPEG-1 Layer 3, while other approaches work on MPEG-1 Layer 1 and 2 also. Finally, another recent work that uses MDCT coefficients from MP3 is [KQG06], they also consider MDCT coefficients from AAC bitstreams. They derive a generic framework for audio classification and segmentation. To our knowledge, this paper is the only work on transform-domain audio indexing that considers the AAC audio coding format.

In this chapter, we consider three audio indexing applications: beat tracking [Sch98, Dix01, KEA06, DP07], chord recognition [Fuj99, SE03, BP05, LS08] and musical genre classification [TC02, BCE+06, HS08]. To our knowledge, beat tracking is the only application that has already been studied in a transform-domain framework, using MP3 coded data [WV01, ZW08]. No work related to chord recognition on coded data have been found in the literature. This may be due to the limited frequency resolution of the time-frequency analysis used in audio coders such as MP3. Due to this limitation of transform audio coders, it is interesting to consider other kinds of audio coders, such as parametric coding (e.g. [dSO02]) or sparse representation based coding. We study here one of our audio codec based on a union of MDCT bases. Our approach has the advantage to provide a sparse representation which has both precise time and frequency resolution, contrary to transform based coders. To compare our proposed audio coder with standard audio coders, we also propose transform-domain systems based on the two most widely used state-of-the-art audio coders, MPEG-1 Layer 3 (MP3) and MPEG-2/4 AAC. For each of the three coders, we propose simple algorithms that compute transform-domain mid-level representations for each of the three applications. A mid-level representation may be defined as an intermediate measure between the low-level PCM signal and the symbolic representation that aims at emphasizing certain structures useful for a given application. In our case, these are respectively a detection function for beat tracking, a chromagram for chord recognition, and MFCC features for musical genre classification. These proposed algorithms are very fast as compared to the reference time-domain implementation. The mid-level representations are then passed through a machine learning system in order to obtain the desired symbolic representation, respectively a sequence of beat positions, a sequence of chords, and a genre class. We use the same machine learning systems as used in state-of-the-art systems in order to compare the mid-level representations only.

The remainder of this chapter is as follows. In section 8.2, we briefly introduce the internal representation of each of the three considered audio codecs. In section 8.3, we propose simple algorithms to compute mid-level representations for each audio codec and for each application. In section 8.4, we describe the three applications, and present the results in terms of performance and computation time.

8.2 Audio coding and transform representations

In this section, we detail the signal representations used in two standard audio codecs MP3 and AAC, and we recall the signal representation used in the proposed codec based on the union of 8 MDCT bases (noted in this chapter "8xMDCT codec"). These signal representations are all based on various forms of MDCT, and thus the transform-domain audio indexing systems we propose in this chapter are all based on the MDCT coefficients of these representations.

8.2.1 MPEG-1 Layer 3

In MPEG-1 Layer 3, a hybrid signal representation is used. The original PCM signal is first passed through a 32-band PQF filterbank (see Sec. 2.2.1). Then, each subband signal is transformed with a time-varying MDCT (see Sec. 2.2.1). The time-varying MDCT is based on two window sizes allowing adaptive time-frequency resolution. The long window has a length of 36 samples and allows a frequency resolution of 38.3 Hz at 44.1 kHz. The short window has a length of 12 samples and allows a time resolution of 4.35 ms at 44.1 kHz. It is important to note that the short windows are always selected by groups of 3 consecutive frames. The same window sequence is used in each subband in order to synchronize the time-varying MDCT of the different subbands (we assume that the mixed block feature is not used). The MDCT coefficients are then grouped in temporal segments called "granule". One granule is composed either by the $32 \times 18 = 576$ coefficients of one frame of a long window MDCT in each subband, either by the $3 \times 32 \times 6 = 576$ coefficients of 3 consecutive frames of a short window MDCT in each subband. The coefficients of a "long-window granule" are noted $X_k^{\text{long}}(q)$ with $k = k_1 18 + k_2$ $(0 \leq k < 576)$ is the frequency index $(0 \leq k_1 < 32$ is the subband index and $0 \leq k_2 < 18$ is the frequency index in one subband), and q is the granule index. The coefficients of a "short-window granule" are noted $X_{p,k}^{\text{short}}(q)$ with $0 \leq p < 3$ is the frame index, $k = k_1 6 + k_2$ $(0 \leq k < 192)$ is the frequency index for one frame $(0 \leq k_1 < 32$ is the subband index and $0 \leq k_2 < 6$ is the frequency index in one subband), and q is the granule index.

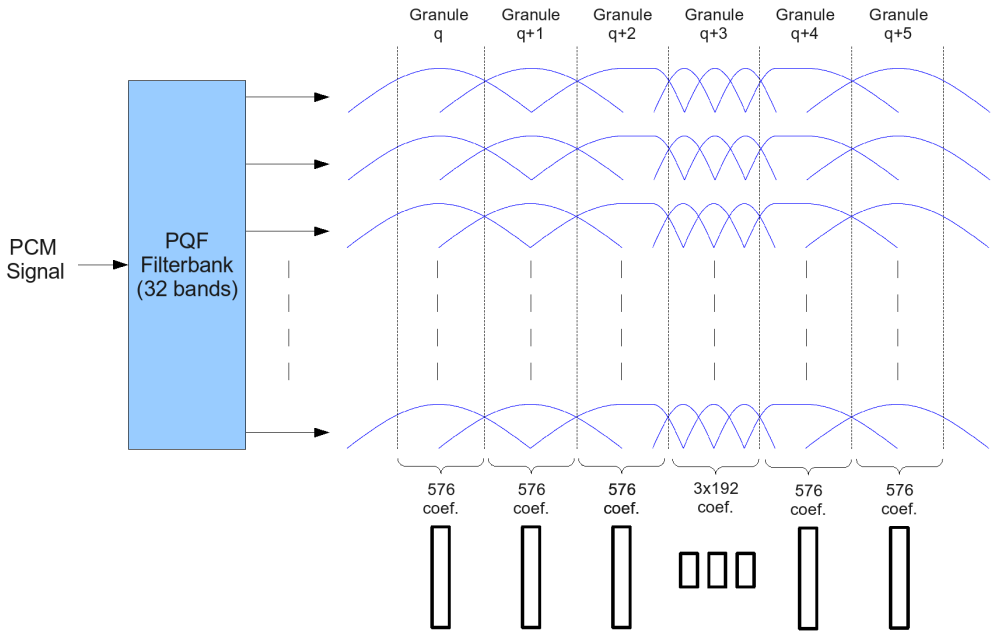

Figure 8.2: *Hybrid signal representation used in MP3*

The resulting MDCT coefficients are then coded using non-linear quantization and Huffman coding. The MP3 decoder simply recovers the MDCT coefficients with Huffman decoding and inverse quantization. We use in our experiment the MP3 encoder LAME [LAM08], a well-known open-source encoder, and the MP3 decoder libMAD [lib08], an open-source library written in C. We had to slightly modify the source of libMAD to be allowed to get the decoded MDCT coefficients instead of the decoded PCM signal.

It is important to note that some existing works (e.g. [TC00]) consider the subband signals instead of the MDCT coefficients to compute transform-domain features. However, doing this has 2 drawbacks, firstly the frequency resolution of the MDCT coefficients is lost, secondly, the computational cost is higher as an inverse MDCT is needed. For these two reasons, we will use in the proposed transform-domain systems the MDCT coefficients only.

8.2.2 MPEG-2/4 Advanced Audio Coding

We use in this chapter the simplest AAC codec, which is the MPEG-4 AAC Low Complexity (LC) profile with the Temporal Noise Shaping (TNS) and Perceptual Noise Substitution (PNS) tools disabled. The signal representation is based on a pure time-varying MDCT and it is thus a simpler approach than MPEG-1 Layer 3. The time-varying MDCT is based on two window sizes. The long window has a length of 2048 samples and allows better frequency resolution than the MP3 long window (21.5 Hz for AAC, 38.3 Hz for MP3 at 44.1 kHz). The short window has a length of 256 samples and allows better time resolution than the MP3 short window (2.90 ms for AAC, 4.35 ms for MP3 at 44.1 kHz). Similarly to MP3, the short windows are selected in groups of 8 consecutive windows in order to produce a group of $8 \times 128 = 1024$ coefficients, which corresponds to the number of coefficients of one frame of a long window. In the case of AAC, one granule is also called a "block", and corresponds to one frame of a long window MDCT or 8 consecutive frames of a short window MDCT. The coefficients of a "long-window block" are noted $X_k^{\text{long}}(q)$, where $0 \leq k < 1024$ is the frequency index and q is the block index. The coefficients of a "short-window block" are noted $X_{p,k}^{\text{short}}(q)$, where $0 \leq p < 8$ is the frame index, $0 \leq k < 128$ is the frequency index for one frame and q is the block index.

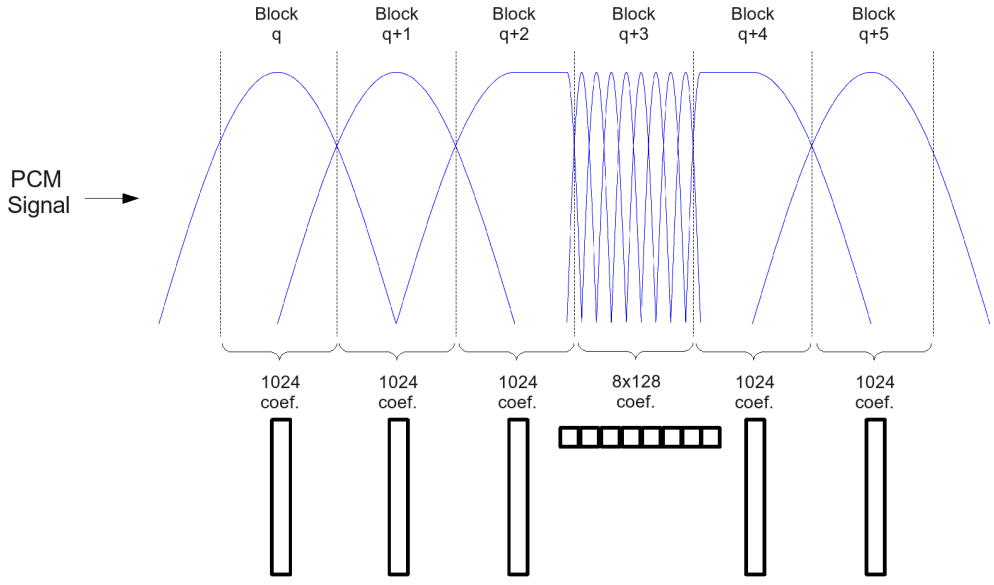

Figure 8.3: *Signal representation used in AAC.*

The resulting MDCT coefficients are then coded using non-linear quantization and Huffman coding. The AAC decoder simply recovers the MDCT coefficients with Huffman decoding and inverse quantization. We use the AAC encoder FAAC [FAA08], an open-source AAC encoder and the AAC decoder FAAD [FAA08], an open-source library written in C. We have slightly modified

the code source of FAAD to be able to get the decoded MDCT coefficients. It is important to note that the AAC encoder we have chosen is not as highly optimized as iTunes or Nero AAC free encoders; however, it is the only coder that is open source and highly configurable. Particularly, it is the only coder that is able to disable the AAC tools such as TNS, and also it is able to produce a raw bitstream that does not need an external MPEG-4 library.

8.2.3 8xMDCT Audio Coding

We recall here briefly the signal representation used in the proposed 8xMDCT coder. The signal is modeled using a union of 8 MDCT bases, where the window length ranges from 128 to 16384 samples (i.e. from 2.9 to 370 ms) in powers of 2. A signal $x \in \mathbb{R}^N$ is then decomposed as a weighted sum of functions $g_\gamma \in \mathbb{R}^N$ plus a residual of negligible energy r

$$x = \sum_{\gamma \in \Gamma} c_\gamma g_\gamma + r \tag{8.1}$$

where c_γ are the coefficients. The set of functions $\mathcal{D} = \{g_\gamma, \gamma \in \Gamma\}$ is the dictionary composed by a union of M MDCT bases (called blocks). The functions g are the atoms defined as:

$$g_{m,p,k}(n) = w_{m,p}(u)\sqrt{\frac{2}{K_m}}\cos\left[\frac{\pi}{K_m}\left(u + \frac{K_m}{2} + \frac{1}{2}\right)\left(k + \frac{1}{2}\right)\right] \tag{8.2}$$

where $u = n - pL_m - T_m$ and m is the block index, p is the frame index, k is the frequency index, K_m is the half of the analysis window length of block m (defined as power of two $K_m = K_0 2^m$), P_m is the number of frames of block m, T_m is a time offset introduced to "align" the windows of different lengths ($T_m = \frac{L_m}{2}$) and $w_m(u)$ is the sine window defined on $u = 0, .., 2L_m - 1$. The signal is approximated with the standard Matching Pursuit, using the Matching Pursuit ToolKit.

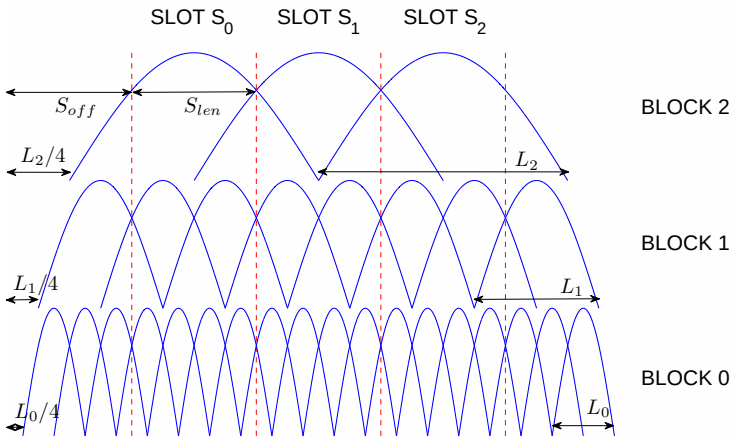

Figure 8.4: *Signal representation used in the 8xMDCT codec (only 3 MDCT are shown).*

8.3 Mid-level signal representations

We propose here several mid-level representations that are computed in the transform domain using the MDCT coefficients of the three coders presented in the previous section. The

basic idea is that most of the standard mid-level representations are based on the Short-Term Fourier Transform (STFT) and thus by substituting the STFT spectrogram by the corresponding MDCT "pseudo-spectrogram", it is possible to compute similar mid-level representations in the transform domain using MDCT coefficients. We are interested here in three types of mid-level representations: onset detection functions, chromagrams, and MFCC-based features. We describe in the following, for each mid-level representation, a reference approach that we use in our experiment, and the proposed transform-domain approaches for each of the three coders.

8.3.1 Onset detection function

Onset detection functions are mid-level representations that aim at localizing transients in an audio signal. These are generally subsampled, and ideally have peaks located at transients. These functions are obviously useful for onset detection, the onsets are simply detected by peak-picking the detection function (see [BP05] for a good review on onset detection algorithms). They are also useful for beat tracking (see e.g. [Sch98, Dix01, KEA06, DP07]), the basic principle is to look for periodically related peaks in the onset detection function, these particular onsets are called "beats". We are interested here in beat tracking. We first review the reference onset detection function we use in our experiments (from [DP07]), then we propose several detection functions computed in the transform-domain and based on the MP3, AAC and 8xMDCT coders.

Reference time-domain onset detection function

The reference onset detection function of [DP07] is the complex spectral difference onset detection function originally proposed by Bello et al. in [BDDS04]. It is the complex extension of the simple spectral difference detection function that is also known as spectral flux. It is defined as

$$\Gamma(q) = \sum_{k=1}^{K} |S_k(q) - \hat{S}_k(q)|^2 \tag{8.3}$$

where $S_k(q) = R_k(q)e^{j\phi_k(q)}$ is the Short-Time Fourier Transform (STFT) of the input signal at frame q and frequency k and $\hat{S}_k(q) = \hat{R}_k(q)e^{j\hat{\phi}_k(q)}$ is the predicted spectrum at frame q and frequency k. The predicted spectrum is defined using the assumption of a steady-state signal, in order that the detection function has low value during steady-state parts and high value at transients. The magnitude of the predicted spectrum is defined as

$$\hat{R}_k(q) = R_k(q-1) \tag{8.4}$$

and its phase is defined as

$$\hat{\phi}_k(q) = \text{princarg}\left(2\phi_k(q-1) - \phi_k(q-2)\right) \tag{8.5}$$

where princarg unwraps the phase value. The STFT uses a hanning window with an overlap of 50 %. In the implementation used, the window has a length of 2048 samples, and the resulting detection function is interpolated by a factor of two in order to get one detection function sample every 11.6 ms at 44.1 kHz. See [BDDS04], [BDA+05] and Appendix A of [DP07] for complete reference.

MP3 transform-domain onset detection function

Ye Wang el al. [WV01, ZW08] propose some onset detection functions built on the MP3 transform-domain. These are simple functions based on the MDCT coefficients energy in subbands. We have implemented and tested these functions, but we have obtained very poor performance using the beat tracking algorithm of M. Davies [DP07]. We think that these functions are optimized for the beat tracking algorithm proposed by Wang et al. in [WV01, ZW08], but not for other algorithms. We obtain much better performance using a detection function similar to the spectral flux. It is is defined as

$$\Gamma(q) = \sum_{k=1}^{576} |X_k(q)^2 - X_k(q-1)^2|^{1/2} \tag{8.6}$$

with $X_k(q)$ is a "pseudo-spectrum" at granule q and frequency k. It is defined for a "long-window granule" as

$$X_k(q) = |X_k^{\text{long}}(q)| \tag{8.7}$$

For a "short-window granule", it is defined as the interleaved coefficients of the 3 short frames

$$X_k(q) = |X_{a,b}^{\text{short}}(q)| \tag{8.8}$$

where a and b are respectively the rest and the quotient of the Euclidean division of k by 3 ($k = 3b + a$). The time resolution is here determined by the granule length and is thus equal to 576 samples (13ms at 44.1 kHz).

AAC transform-domain onset detection function

We propose a transform-domain onset detection function based on the AAC MDCT coefficients that is similar to the proposed MP3 transform-domain onset detection function. A "pseudo-spectrum" is first computed using the MDCT coefficients, it is defined as

$$X_k(q) = |X_k^{\text{long}}(q)| \tag{8.9}$$

for a long-window block and as

$$X_k(q) = |X_{a,b}^{\text{short}}(q)| \tag{8.10}$$

for a short-window block, where a and b are respectively the rest and the quotient of the Euclidean division of k by 8 ($k = 8b + a$). The detection function is then defined as

$$\Gamma(q) = \sum_{k=1}^{576} |X_k(q)^2 - X_k(q-1)^2|^{1/2} \tag{8.11}$$

In this case, the time-resolution is higher and equal to the block length i.e. 1024 samples. To get the same sample rate as the reference approach, the detection function is interpolated by a factor of two, resulting in one sample every 11.6 ms at 44.1 kHz.

8xMDCT transform-domain onset detection function

The signal representation used in our proposed audio coder is based on a union of 8 MDCT bases with analysis window sizes from 128 to 16384 samples. We have remarked that high amplitude atoms with small window sizes (128 and 256) are often located around attacks; consequently, we can build a very simple onset detection function by sorting the decomposition such that we keep

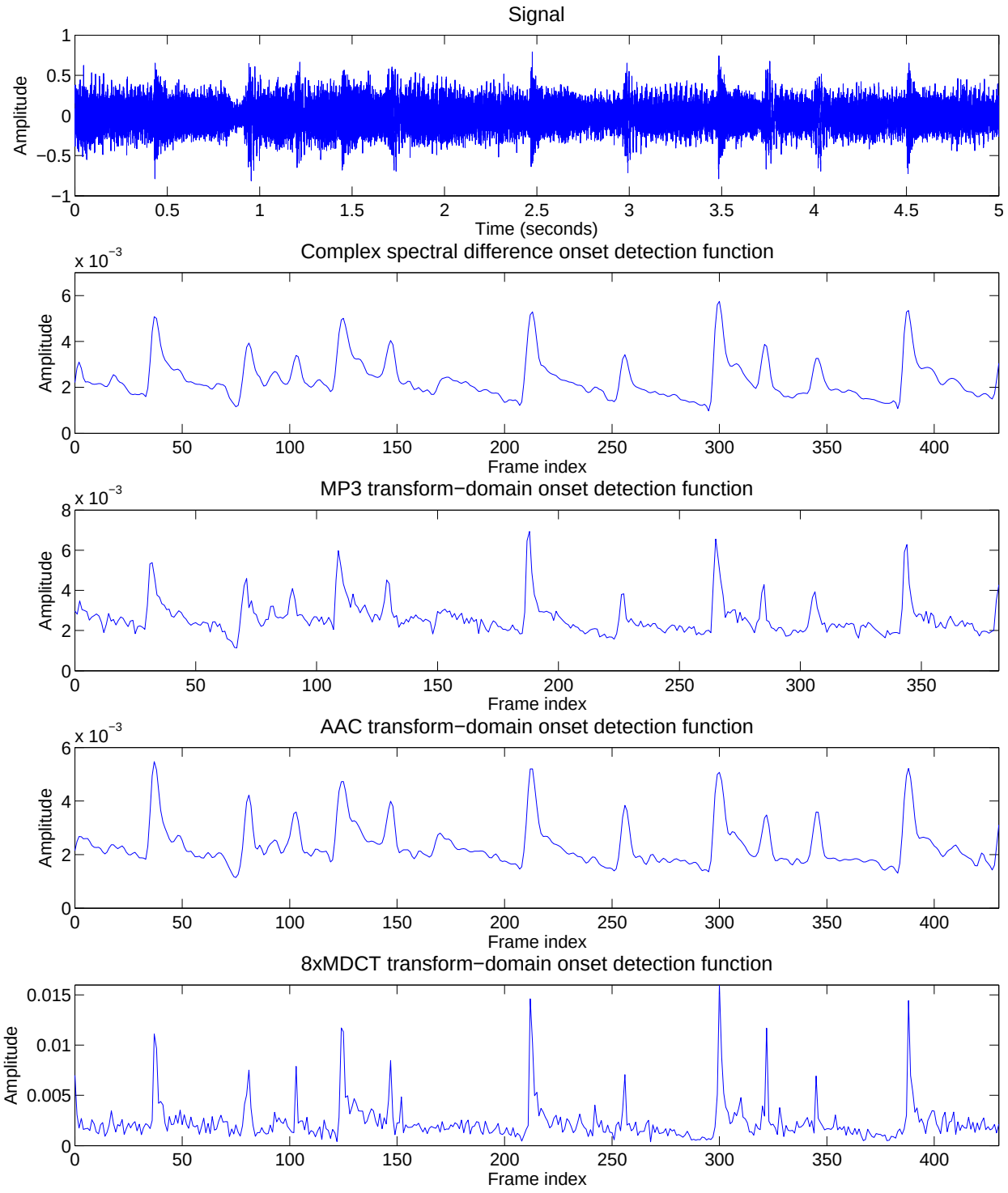

Figure 8.5: *A 5 seconds signal of rock music; the complex spectral difference onset detection function; the MP3 transform-domain onset detection function; the AAC transform-domain onset detection function; the 8xMDCT transform-domain onset detection function.*

only small window sizes atoms, and then sum the absolute value of the coefficients in temporal bins to construct a downsampled signal with peaks located at attacks. The proposed onset detection function Γ is computed on a frame-by-frame basis. The length of one frame is defined such that the corresponding time resolution is the same as in the reference detection function which is 11.6 ms and it is equivalent to $t_{DF} = 512$ samples at 44.1 kHz. The function $\Gamma(q)$ at frame q is defined as

$$\Gamma(q) = \sum_{m,p,k} |c_{m,p,k}| \tag{8.12}$$

where we sum only the atoms satisfying the following two conditions:

- the window size is 128 or 256 samples
$$m < 2 \tag{8.13}$$
- the center is in the temporal support of the frame q
$$floor\left(\frac{(p+1)L_m + T_m}{t_{DF}}\right) = q. \tag{8.14}$$

Fig. 8.5 shows the four onset detection functions obtained with a 5-second signal of rock music. In this example, the onset detection functions have peaks that correspond to the drum strokes.

8.3.2 Chromagram

A chromagram or Pitch Class Profile (PCP) [Fuj99] traditionally consists of a 12-dimensional vector, with each dimension corresponding to the intensity of a semitone class (chroma). The procedure collapses pure tones of the same pitch class, independent of octave, to the same chromagram bin; for complex tones, the harmonics also fall into particular related bins. Though, the simplest way is to use a 12-bin chromagram, better modeling is obtained by using more bins (24 or 36), in order to obtain better resolution and compensate for possible mis-tuning. These features find obvious interests in tonality-related applications, such as key estimation [Pau04, GH04] and chord recognition [Fuj99, SE03, BP05, LS08]. We are interested here in chord recognition. We first review the reference chromagram we use in our experiments (from [BP05]), then we discuss how to build transform-domain chromagrams.

Reference time-domain chromagram

The reference chromagram is based on the constant-Q transform [Bro91] and it is the chromagram used in [BP05]. It is defined, at frame q and frequency bin b ($0 \leq b < B$), as

$$CH(q,b) = \sum_{\gamma=1}^{\Gamma} X^{CQ}(q, b + \gamma B) \tag{8.15}$$

where $X^{CQ}(q,k)$ is the constant-Q transform at frame q and frequency $f_k = 2^{k/B} f_{min}$, where B is the number of bins per octave, f_{min} is the starting point of the analysis in frequency and Γ is the total number of octaves. The constant-Q transform is based on the STFT of the signal, using the method proposed in [BP92]. In our implementation, the signal is first downsampled to 11.025 kHz, then a STFT with a window length of 8192 samples is applied, resulting in a frequency resolution of 1.35 Hz.

MP3/AAC transform-domain chromagram

One possibility to build a chromagram from MP3/AAC transform-domain representations is to map the MDCT coefficients to chroma-related frequency bin. However, the frequency analysis performed by the MDCT used in MP3 and AAC does not have enough frequency resolution for an efficient chromagram. Indeed, with a resolution as high as 21.5 Hz for AAC and 38.3 Hz for MP3, the system can't distinguish neighboring notes, and this is particularly true at low frequencies.

Despite this limitation, we have implemented a chromagram based on a single MDCT, with a fixed window size of 2048 samples, which is the best case and corresponds to a long-window block of AAC, and we will show later that very poor performance is obtained with this chromagram.

It is important to note that better frequency resolution may be obtained by using more complex approaches such as the one proposed in [MD03], where parameters of a stationary sinusoid are estimated using the MDCT coefficients. However, this approach was designed for monophonic signals, and it is not clear how it would be generalized to polyphonic signals. Moreover, this transform-domain approach requires rather high computational complexity. Consequently, we have not investigated this approach.

8xMDCT transform-domain chromagram

Contrary to the MDCT used in AAC and MP3, the 8xMDCT codec uses much longer window lengths (up to 16384 samples), and thus allow much better frequency resolution than the MP3/AAC MDCT (up to 2.7 Hz). Consequently, we found it possible in this case to build an efficient transform-domain chromagram. The basic principle is to sort the decomposition such that we only keep the atoms with good frequency resolution (i.e. with high window length), and then sum the absolute value of the coefficients of these atoms in chroma-related time/frequency bins. In our implementation, we have decided to keep the two largest window sizes (8192 and 16384). However, it is important to note that the frequency resolution is still lower than the representation used in the reference chromagram (1.3 Hz), and we will see later that this results in slightly lower performance as compared to the reference system. The 8xMDCT transform-domain chromagram $CH(q,b)$ at frame q and frequency bin b is defined as

$$CH(q,b) = \sum_{m,p,k} |c_{m,p,k}| \tag{8.16}$$

where we only sum the atoms satisfying the following three conditions:

- the window size is 8192 or 16384 samples

$$m \geq 6 \tag{8.17}$$

- the center is in the temporal support of the frame q

$$floor\left(\frac{(p+1)L_m + T_m}{t_{CH}}\right) = q. \tag{8.18}$$

- the frequency value k maps to the frequency bin b of the chromagram

$$\mathrm{mod}\left(\mathrm{round}\left(B\ \mathrm{log2}\left(\frac{22050\ \mathrm{k}/L_m}{f_{min}}\right)\right), B\right) = b \tag{8.19}$$

  with B the number of bins per octave and f_{min} is the minimum frequency.

Fig. 8.6 shows the chromagrams obtained with a 50 seconds signal of rock music.

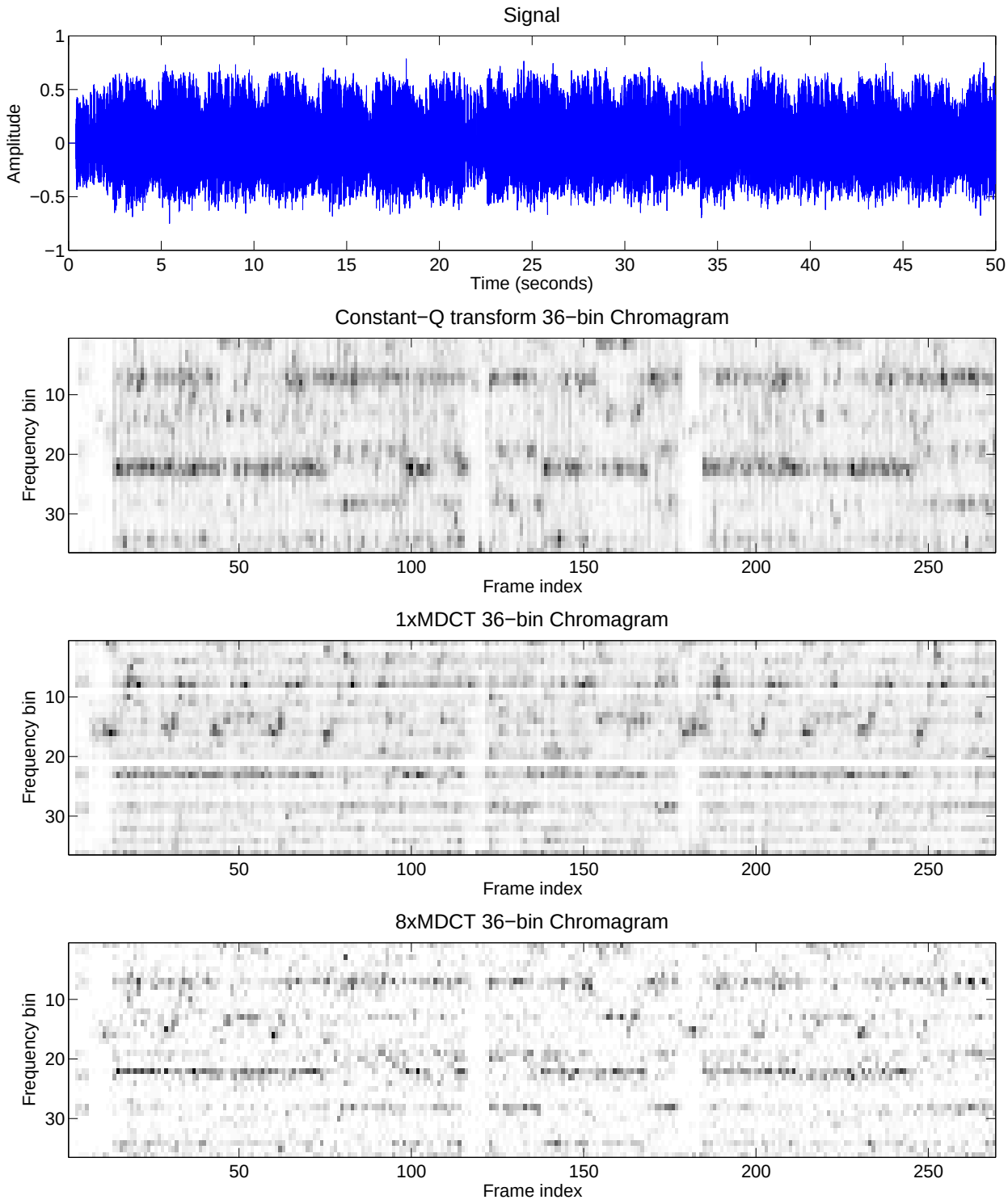

Figure 8.6: *A 50 seconds signal of rock music; the reference chromagram; the proposed chromagram based on a single MDCT with window length* 2048 *samples; the proposed transform-domain chromagram based on the 8xMDCT coder.*

8.3.3 Mel-Frequency Cepstrum Coefficients

Mel-Frequency Cepstrum Coefficients (MFCC) aims at providing a compact representation of the spectral envelope of an audio signal. These features were originally developed for speech recognition [DM80], as they model the vocal tract transfer function. They are now widely used in musical applications, as they appear to be a good description of the timbre. They find useful applications in e.g. musical genre classification [TC02] and music similarity [PA04]. More recently, MFCC are used in baseline systems for evaluating audio classification systems (e.g. [LLV+08, HS08]).

Reference time-domain MFCC

We use the implementation of the MA toolbox by E. Pampalk [Pam06]. The computation of a set of C MFCC coefficients is described as follows. A small frame $x(n), n = 0, .., N-1$ is first extracted from the signal. In our implementation, the frames are non-overlapping and have a length of 23.2 ms i.e. $N = 1024$ samples at 44.1 kHz. Then, the magnitude of the Discrete Fourier Transform is computed

$$X(k) = |\sum_{n=0}^{N-1} x(n)w(n)e^{-ikn/N}| \ , \ k = 0, .., N/2-1 \tag{8.20}$$

with $w(n), n = 0, .., N-1$ the analysis window. In our implementation, it is a Hamming window. Then, the resulting spectrum $X(k)$ is mapped onto the Mel scale using L triangular overlapping windows that are equally spaced on the Mel scale. In our implementation, we use $L = 40$ triangular windows whose frequency bounds range from 20 Hz to 20000 Hz, and we use the following formula for the Mel-scale:

$$m = 1127.01048 \log(1 + f/700) \tag{8.21}$$

where m is the frequency in mel and f is the frequency in Hz. Fig. 8.7 shows the 40 triangular overlapping windows.

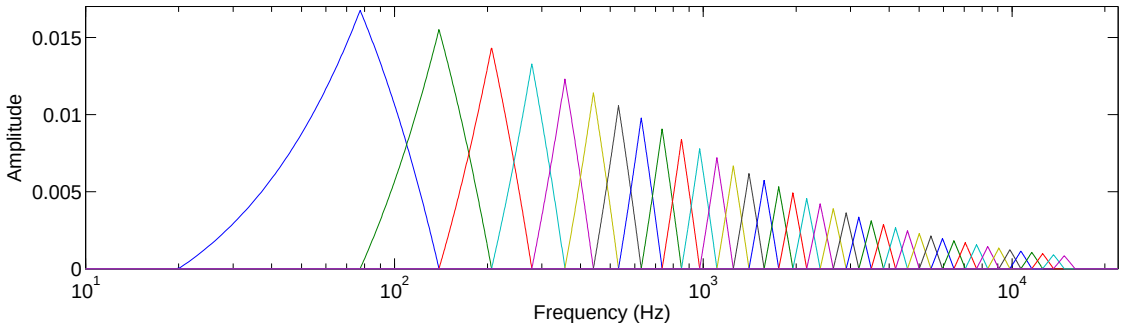

Figure 8.7: *The 40 triangular overlapping windows equally spaced on the Mel scale.*

The mapped spectrum $Y(l), l = 0, .., L-1$ is then defined as

$$Y(l) = \sum_{k=0}^{N/2-1} X(k)W_l(k) \ , \ l = 0, .., L-1 \tag{8.22}$$

where $W_l(k), k = 0, ..., N/2 - 1$ is the l-th window. Finally, the mapped spectrum is converted to dB and transformed with a Discrete Cosine Transform. The final MFCC coefficients are defined as

$$mfcc(c) = \frac{1}{L/2}\Lambda(c)\sum_{l=0}^{L-1} 10\log10(Y(l) + \text{eps})\cos\left(\frac{\pi}{L}\left(l + \frac{1}{2}\right)c\right) \ , c = 0, ..., C - 1 \tag{8.23}$$

with $\Lambda(c) = \sqrt{2}/2$ if $c = 0$ and $\Lambda(c) = 1$ otherwise. The constant $\text{eps} = 1e - 16$ avoids log of zero. In our implementation, we keep $C = 13$ coefficients. The complete algorithm is summarized below

1. Take a N-length frame of an audio signal.
2. Compute the DFT of the windowed frame.
3. Take the magnitude of the DFT output.
4. Map to Mel scale using 40 overlapping triangular windows.
5. Convert to dB.
6. Transform with DCT.

MP3/AAC transform-domain MFCC

We propose here a simple algorithm for the computation of a set of MFCC in the transform-domain of MP3/AAC audio files. The basic principle is to use the absolute value of the MDCT coefficients instead of the magnitude of the DFT in the MFCC computation described previously. The rest of the algorithm is exactly the same. It is important to note that the MFCC are computed on long-window blocks only (on MDCT-coefficient vectors of length 576 for MP3 and 1024 for AAC) for two reasons, firstly the frequency resolution of small-window blocks is too low, secondly we want to have comparable feature vectors in order to estimate long-term statistics (see later "texture window"). The algorithm is summarized below

1. Take a 576/1024-coefficient vector of a long-window block.
2. Take the absolute value of the coefficients.
3. Map to Mel scale using 40 overlapping triangular windows.
4. Convert to dB.
5. Transform with DCT.

8xMDCT transform-domain MFCC

We propose here an algorithm to compute MFCC-like features from the transform-domain representation of the 8xMDCT codec. This has been inspired by the work of M. Morvidone (reference not published yet) who works on automatic identification of musical instruments. These features are computed on a frame-by-frame basis, where a vector of features is computed for each frame of 8192 samples (this is equal to the hop size of the maximum window length, and it is equivalent to the "timeslots" used in audio coding). In each frame, a scale-frequency representation is computed, where the frequency axis is on the same Mel-scale as in the reference MFCC computation, and the scale axis corresponds to the window size. This representation can be seen as

scale-dependant MFCC. This scale-frequency representation is simply a weighted histogram where the amplitude of the atoms are summed in scale-frequency bins. The scale-frequency representation $Y(l,m)$ is defined as

$$Y(l,m) = \sum_{p,k} |c_{m,p,k}| W_{m,l}(k) \ , \ l = 0,..,L-1 \ , \ m = 0,..,M-1 \tag{8.24}$$

where $W_{m,l}(k)$ is the l-th window of the scale m. This scale-frequency representation is then converted to dB and transformed with a 2D-DCT. The final MFCC coefficients are defined as

$$mfcc(i,j) = \frac{1}{L/2}\frac{1}{M/2}\Lambda(c_i)\Lambda(c_j)\sum_{l=0}^{L-1}\sum_{m=0}^{M-1} Y^{DB}(l,m)\cos\left(\frac{\pi}{L}\left(l+\frac{1}{2}\right)c_i\right)\cos\left(\frac{\pi}{M}\left(m+\frac{1}{2}\right)c_j\right) \tag{8.25}$$

with $i = 0,..,C_j - 1$, $j = 0,..,J-1$. The total number of MFCC coefficients is then equal to $C = \sum_{j=0}^{J-1} C_j$. In our implementation, we choose $J = 4$, and $C_0 = 7$ coefficients on the first scale axis, $C_1 = 3$ on the second scale axis, $C_2 = 2$ on the third scale axis and $C_3 = 1$ on the fourth scale axis. This results in a total number of coefficients $C = 13$.

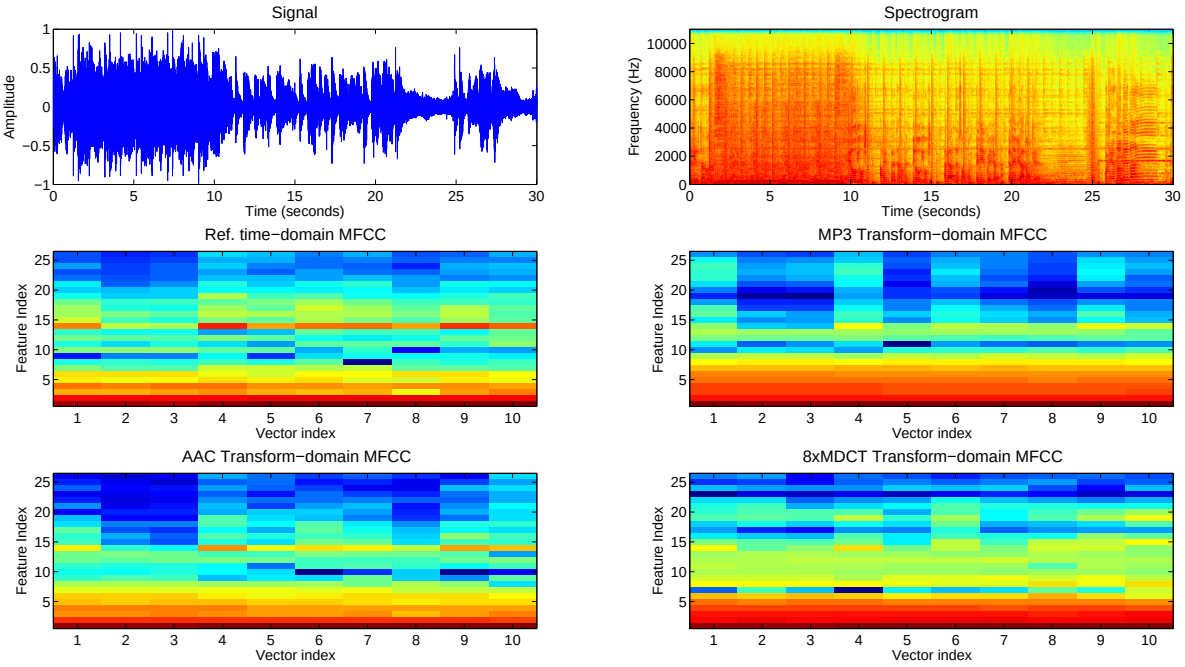

Figure 8.8: *A 30-second signal of rock music; the spectrogram; the reference time-domain MFCC and the 3 transform-domain MFCC (mean and variance for each texture window).*

Texture window

MFCC are computed on segments of length 23.2 ms for the reference implementation, 13.0 ms for MP3, 23.2 ms for AAC, and 185.8 ms for 8xMDCT coding. As proposed in [TC02, BCE+06, HS08], the MFCC are grouped in longer frames, also called texture windows. In our implementation, we take the mean and the variance of the MFCC on texture windows of length 3 seconds. This results in a vector of 26 features for each 3 seconds of an audio signal. Fig. 8.8 shows a 30-second signal of rock music, its spectrogram, and the different MFCC implementations.

8.4 Applications

We have proposed in the previous section three mid-level representations that can be computed in the transform-domain, respectively an onset detection function, a chromagram and MFCC features. We now integrate these mid-level representations in complete audio indexing systems. For each of the three considered applications, respectively beat tracking, chord recognition and musical genre classification, we describe the audio indexing system, then evaluate the performance and finally give computation time results.

8.4.1 Beat tracking

Machine learning system The same system as in [DP07] is used (see Fig. 8.9). The onset detection function is first post-processed using an adaptive moving average threshold. Then the onset detection function is partitioned into overlapping frames to allow variable tempo. In each frame, the unbiased autocorrelation function of the onset detection function is calculated. The autocorrelation function is then passed into a shift-invariant context-dependant comb filterbank in order to estimate the tempo of the current frame. Finally, a beat train at the estimated tempo is built and aligned with the current frame by passing the detection function into a tuned context-dependant comb filterbank.

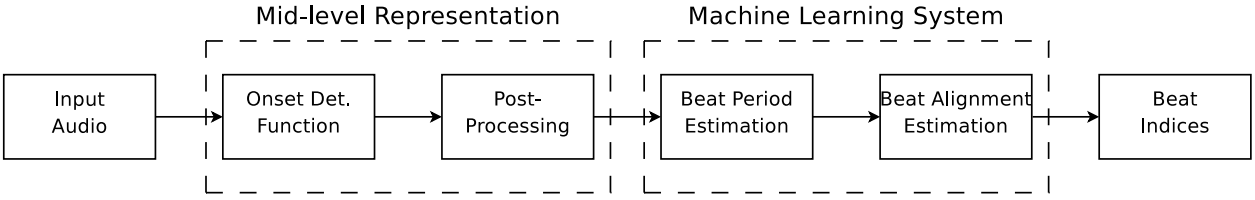

Figure 8.9: *Beat tracking system of M. Davies [DP07].*

Evaluation metrics We give four measures of beat accuracy, as proposed in [KEA06] and used also in [DP07]: correct metrical level with continuity required (CML cont); correct metrical level with continuity not required (CML total); allowed metrical levels with continuity required (AML cont); allowed metrical levels with continuity not required (AML total). See given references for details on these measures.

Evaluation database We use the same database as used in [DP07], which was originally provided by S. Hainsworth [Hai04]. There are 222 files of several music genres. The files are mono, sampled at 44.1 kHz and have a length of approximately 60 seconds. The database was annotated by a trained musician, by recordings taps in time to the audio recordings. The annotations were then corrected and refined using synthesized beat sounds over each track (see [Hai04] for details).

Results Fig. 8.10 and Fig. 8.11 show the obtained performance of the reference and the proposed transform-domain systems at 32kbps and 64kbps. We give the results for two bitrates only due to the limitations of the FAAC encoder (allowed bitrate between 30 and 76 kbps). These results show that the transform-domain systems have a performance close to the reference system. They also show that the performance is robust against the bitrate. This is particularly true for MP3, whose

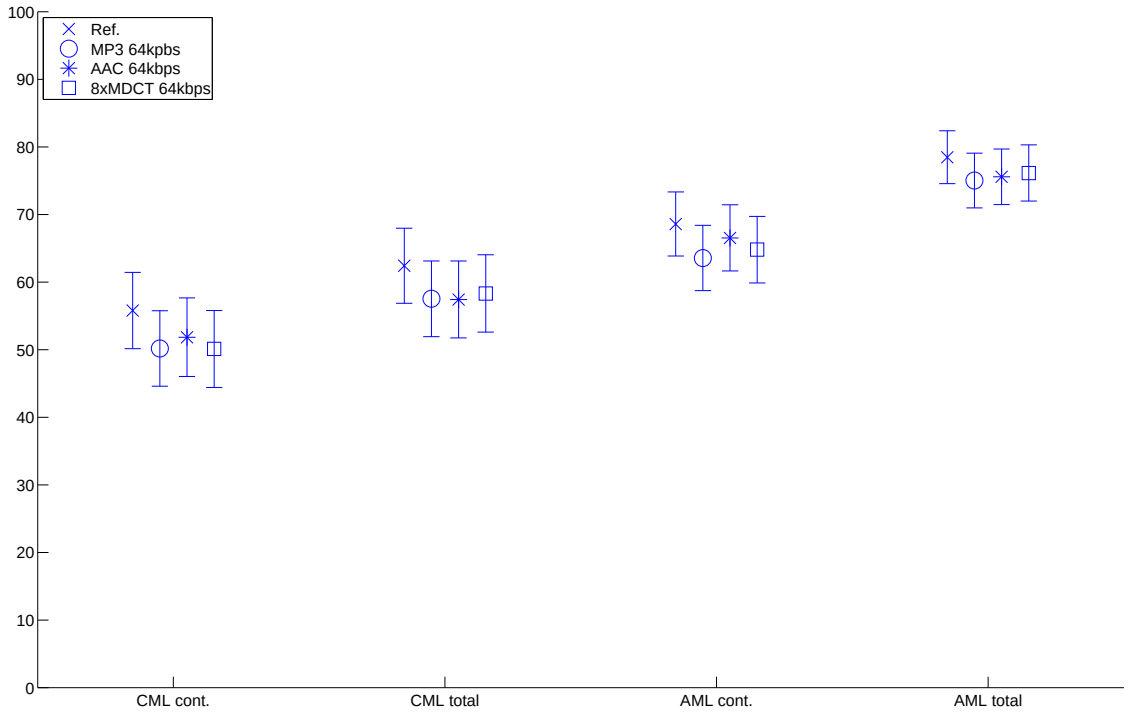

Figure 8.10: *Mean and 95% confidence interval of the 4 beat accuracy measures for the reference and the proposed transform-domain beat tracking systems at 64 kbps.*

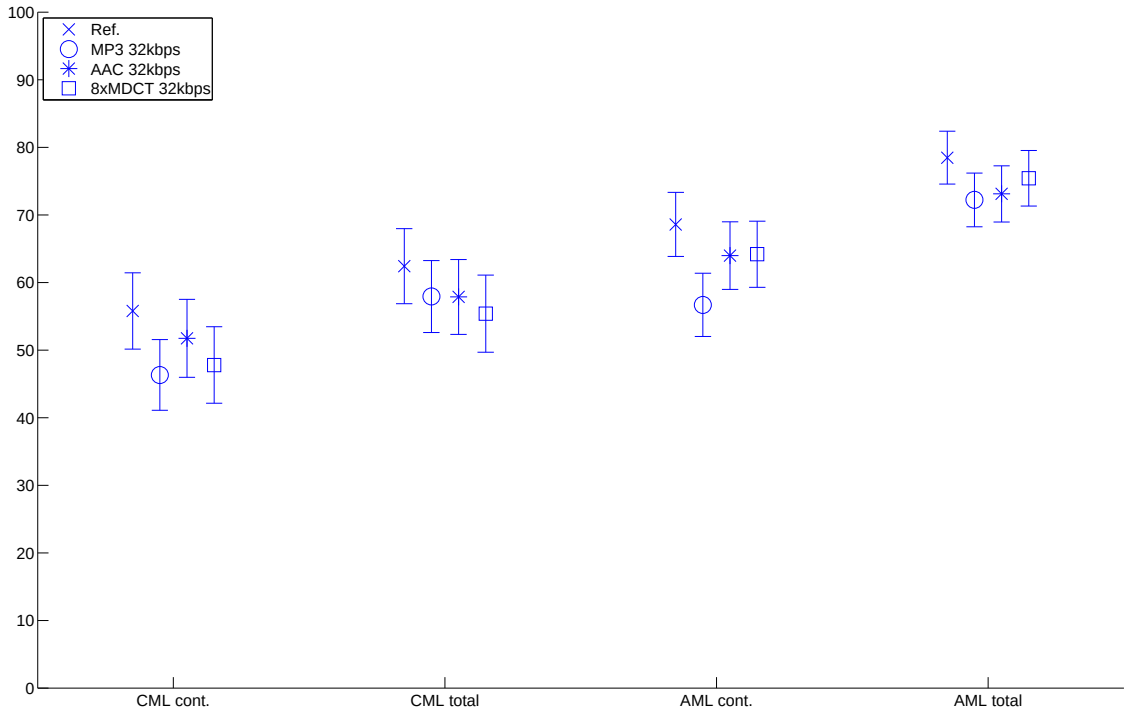

Figure 8.11: *Mean and 95% confidence interval of the 4 beat accuracy measures for the reference and the proposed transform-domain beat tracking systems at 32 kbps.*

quality is bad at 32kbps. Finally it shows that the transform-domain systems give similar results, no one seems to outperform the others.

As our 8xMDCT codec is a scalable coder that has no bitrate limitations, we give the performance obtained by our coder on a wider range of bitrates. Results are in Fig. 8.12. These results show that the performance of the transform-domain beat tracking system based on the 8xMDCT is highly robust against the bitrate. Even at 8kbps, with a very bad audio quality, the performance is high.

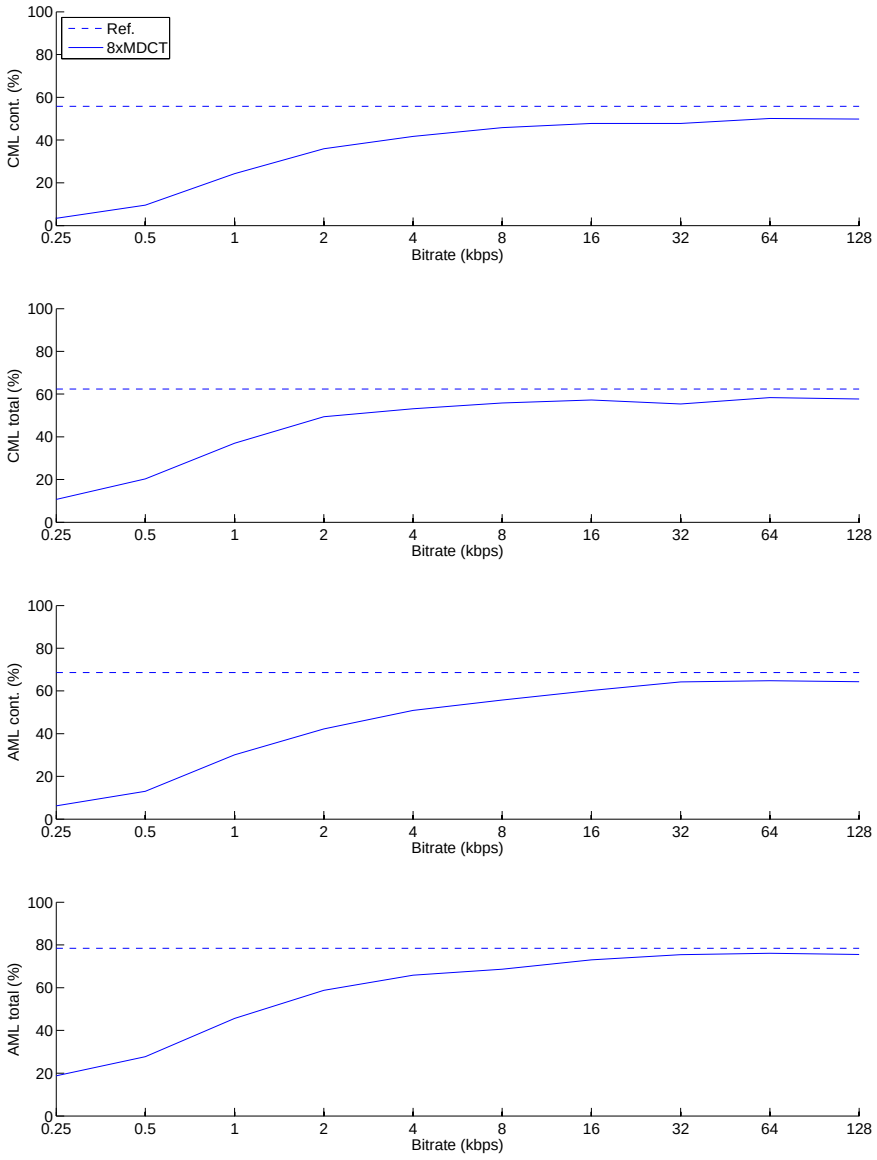

Figure 8.12: *Mean of the 4 beat accuracy measures for the reference and the proposed beat tracking system.*

8.4.2 Chord recognition

Machine learning system The same system as in [BP05] is used (see Fig. 8.13). The 36-bin chromagram is first circularly shifted according to the estimated tuning of the piece, low-pass filtered, and mapped to a 12-bin chromagram by simply summing within semitones. Then, the Expectation Maximization (EM) algorithm is used to train the initial states probabilities and the transition matrix of an Hidden Markov Model (HMM). Finally, the sequence of chords is estimated using the Viterbi algorithm with the chromagram and the trained HMM. The system recognizes 24 chords only (C major, C minor, C# major, C# minor...).

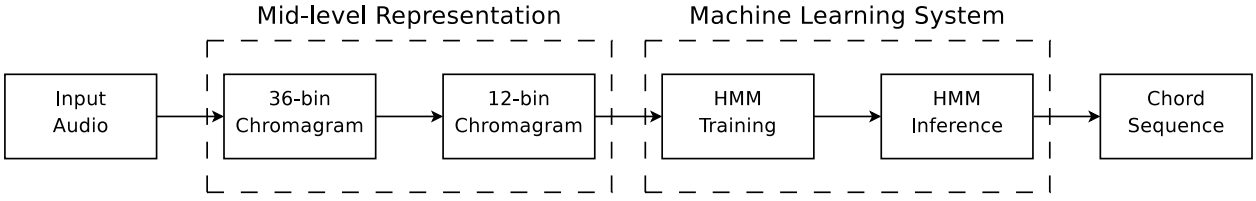

Figure 8.13: *Chord recognition system of J. P. Bello et al [BP05].*

Evaluation database We use the same evaluation database as in [BP05]. It consists of 2 albums of the Beatles: Please Please Me (14 songs) and Beatles for Sale (14 songs). Audio signal are mono and sampled at 44.1 kHz. The database has been annotated by C. Harte et al [HS05]. As some chords in the database do not belong to the set of 24 recognized chords, these complex chords are mapped to their root triad as explained in [BP05].

Evaluation metric We use a simple metric to evaluate the chord recognition systems which is the percentage of well detected frames.

Results Fig. 8.14 shows the obtained performance of the reference and the proposed transform-domain system based on the 8xMDCT coder. As explained in the previous section, we don't give results for the MP3 and AAC codecs due to the limited frequency resolution of their representations. Instead, we give results obtained with a single MDCT with window size 2048 samples, which corresponds to a long-window block of AAC, and is thus the best case i.e. with the best frequency resolution. These results show that the transform-domain system based on the 8xMDCT codec has a performance close to the reference system at high bitrates. It also shows that the performance is highly robust against the bitrate. Even at 2kbps, with a very bad audio quality, the performance is high. Finally, these results show the bad performance obtained with the single MDCT and confirms the limited frequency resolution of the MP3 and AAC codecs explained in the previous chapter.

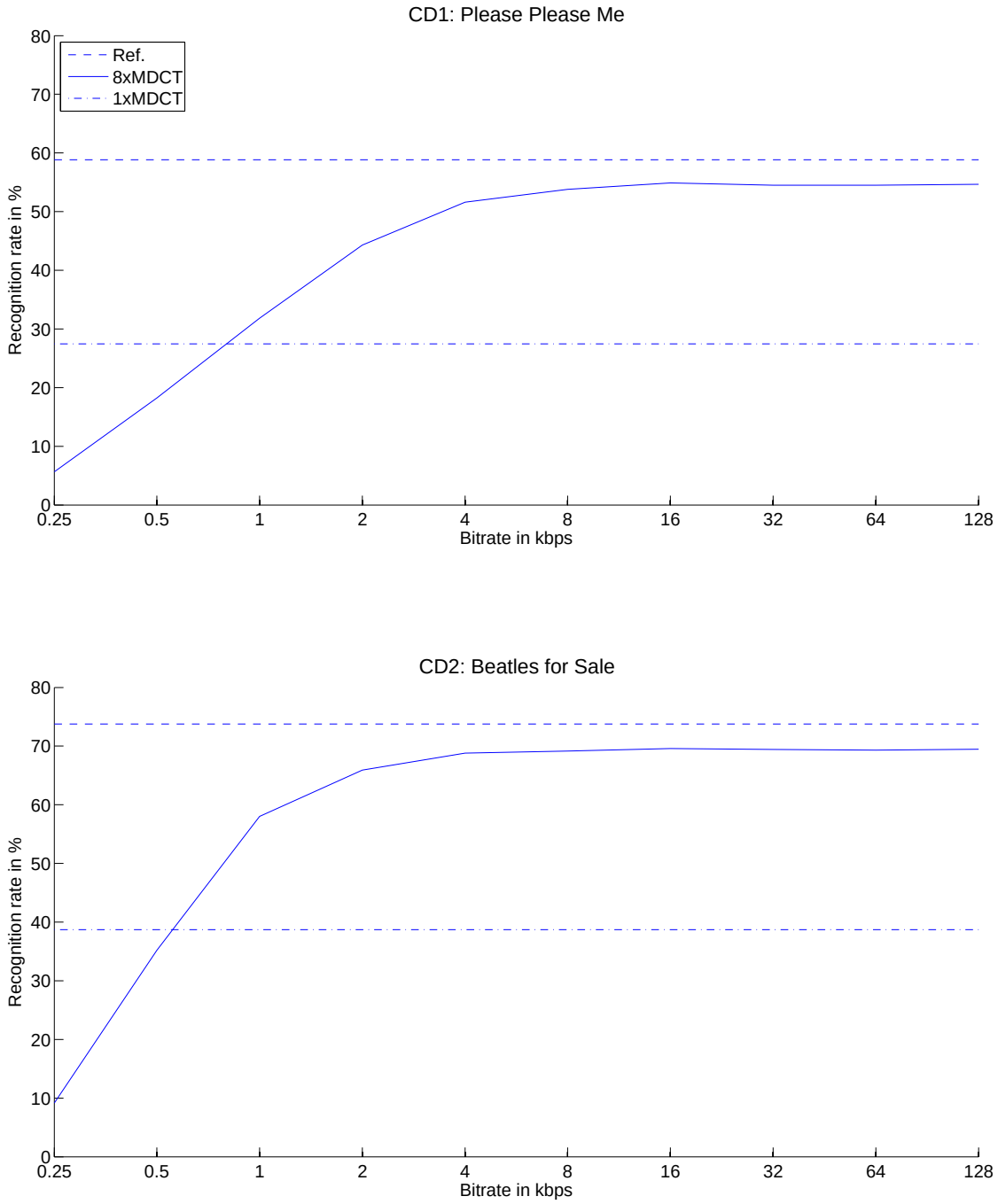

Figure 8.14: *Performance of the proposed chord recognition systems in function of the codec bitrate. The dotted line corresponds to the performance of the reference system on the original PCM audio; and the dash-dotted line corresponds to the performance of the proposed system with a single MDCT (length* 2048 *samples).*

8.4.3 Musical genre classification

Machine learning system A simple system is used (see Fig. 8.15). MFCC are first computed on temporal segments, then the mean and the variance of the coefficients are computed on longer frames called texture windows (see previous section). The length of the texture window is 3 seconds, and the number of MFCC is 13. There are then 26 features for each 3 seconds of audio signal. As an example, for a 30-second signal, there are 10 vector of 26 features. A SVM classifier is used to classify each vector in a genre class. We use libSVM, a high-performance and easy to use open source library. Finally, each vector votes for a genre class, and the class with the maximum number of votes is attributed to the whole song.

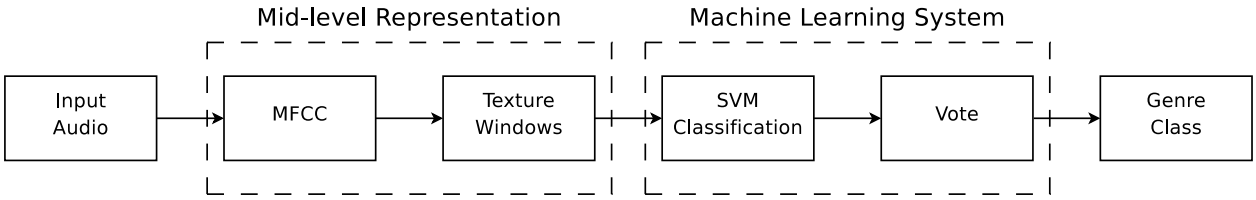

Figure 8.15: *Proposed genre classification system.*

Databases We evaluate the musical genre classification systems on one database. It is one of the two databases used in [HS08]. It is a database originally provided by G. Tzanetakis [TC02]. It is composed by 1000 tracks classified in 10 genres (blues, classical, country, disco, hiphop, jazz, metal, pop, reggae, rock), where each genre class contains 100 tracks. The tracks are mono, 30-second length and sampled at 22.05 kHz. As we have designed our coder for signals sampled at 44.1 kHz only, the tracks are resampled at 44.1 kHz.

Evaluation metric The standard 5-fold cross-validation procedure is used (as in [HS08]). The dataset is first randomly partitioned into 5 equal-size subsets. Then, 4 subsets are chosen to train the SVM model, and the remaining subset is evaluated using the train model. This procedure is repeated 5 times such that all subsets have been evaluated. The classification accuracy is then the total number of good classification. To avoid biased results due to the random partitioning, the cross-validation procedure is repeated 100 times and the final result is the mean of the 100 classification accuracies.

Results Results are given in Fig. 8.16 and Fig. 8.17. These results show that all system achieved good results, with a preference for the proposed codec which outperforms the MP3 and AAC systems and obtain performance similar to the reference system.

8.4.4 Computation times

As stated previously, the performance of the transform-domain systems is similar or slightly lower than the one of the reference systems. However, working in the transform domain allows a huge gain in computation times. As we are working with coded audio, there are two possibilities for a user to calculate mid-level representations from the coded audio. The first possibility is to decode the bitstream, synthesize the decoded transform representation, and calculate state-of-the-art mid-level representations on the synthesized PCM audio. This is the best approach in

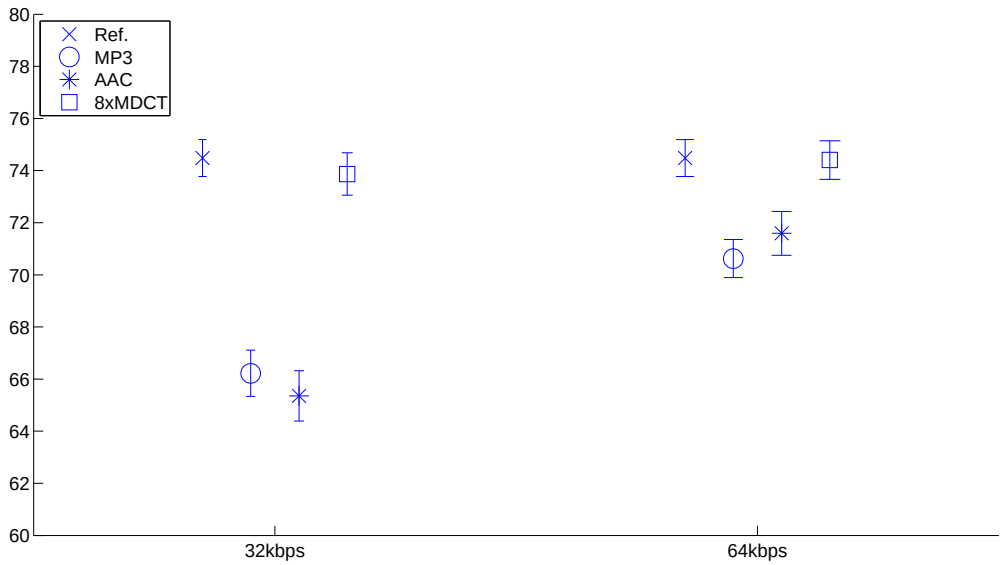

Figure 8.16: *Classification accuracy for the* 4 *musical genre classification systems at* 32 *and* 64 *kbps.*

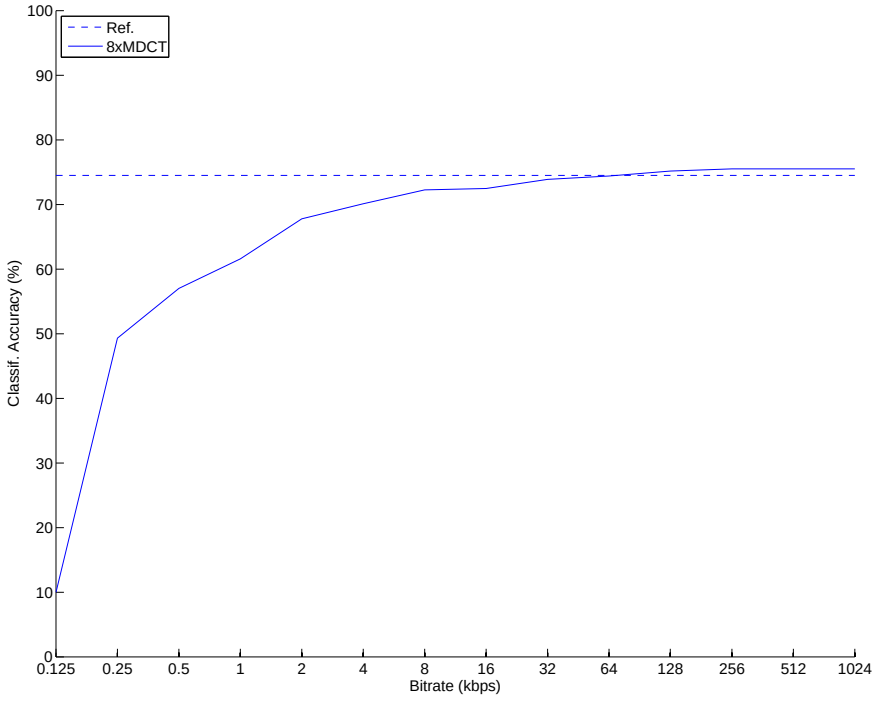

Figure 8.17: *Classification accuracy for the reference and the 8xMDCT musical genre classification systems in function of the bitrate.*

terms of performance. However, it requires several operations with a non negligible computational cost: the bitstream decoding; the transform representation synthesis; the mid-level representation. The second possibility is to compute mid-level representations from the transform representation as explained in the previous sections. This approach is a bit less efficient in terms of performance but it requires less operations that the first approach and thus is faster. Only two operations are required: the bitstream decoding; and the mid-level representation calculation, which has small computation cost as compared to the bitstream decoding cost. We detail below the computation times of each operation, for each codec and each application. It is important to note that some operations are fast implementations in C (MP3/AAC/8xMDCT decoding and synthesis, transform-domain representations, SVM classifier), while others are slow Matlab implementations (reference time-domain mid-level representations, beat tracker, HMM classifier). The reason is that we wanted to keep untouched the reference systems provided by M. Davies, J. Bello, and E. Pampalk, which are implemented in Matlab. All other module have been implemented in C. Consequently, the results are biased by these different implementations, but the general behavior remains the same.

Fig. 8.18, Fig. 8.19 and Fig. 8.20 show the normalized computation times obtained by the time-domain and transform-domain systems, for each codec and for each application. The figures shows the computation times of the different operations: the bitstream decoding; the synthesis; the mid-level representation; and the machine learning system. We detail the results below.

MP3 In the case of MP3, the bitstream decoding is very fast, this is due to the highly optimized library used (libMAD). Moreover, the transform-domain mid-level representation and the beat tracker are very fast operations too. This has for consequence a huge saving in computation time as compared to the time-domain systems. The beat tracking transform-domain system is 12.5 times faster than the time-domain system, while the musical genre classification transform-domain system is 11.5 times faster than the time-domain system.

AAC As compared to MP3, the bitstream decoding is a bit more costly, while the other operations are approximately the same. Consequently, the AAC systems are slightly slower as compared to the MP3 systems. But the transform-domain systems still remain fast. The beat tracking transform-domain system is 10 times faster than the time-domain system, while the musical genre classification transform-domain system is 8 times faster than the time-domain system.

8xMDCT The results are in this case significantly different from the MP3/AAC systems because the bitstream decoding is here much slower. Moreover, the decoding cost depends here on the bitrate. As an example, the bitstream decoding at 2 kbps is 3 times faster than at 64 kbps, but it is still 8 times slower than the MP3 bitstream decoding. Consequently, the transform-domain systems based on 8xMDCT have the advantage to be scalable in complexity, but the drawback to be slower than the MP3/AAC systems. At 64kbps, the beat tracking transform-domain system is 4.5 times faster than the time-domain system, the chord recognition transform-domain system is 2.5 times faster than the time-domain system, and the musical genre classification transform-domain system is 3.5 times faster than the time-domain system.

It is also interesting to remark than the gain in computation times for the transform-domain systems is even higher if several applications are computed at the same time. For example, considering the 8xMDCT codec at 64 kbps and the 3 applications, the transform-domain system is 5 times faster than the time-domain system.

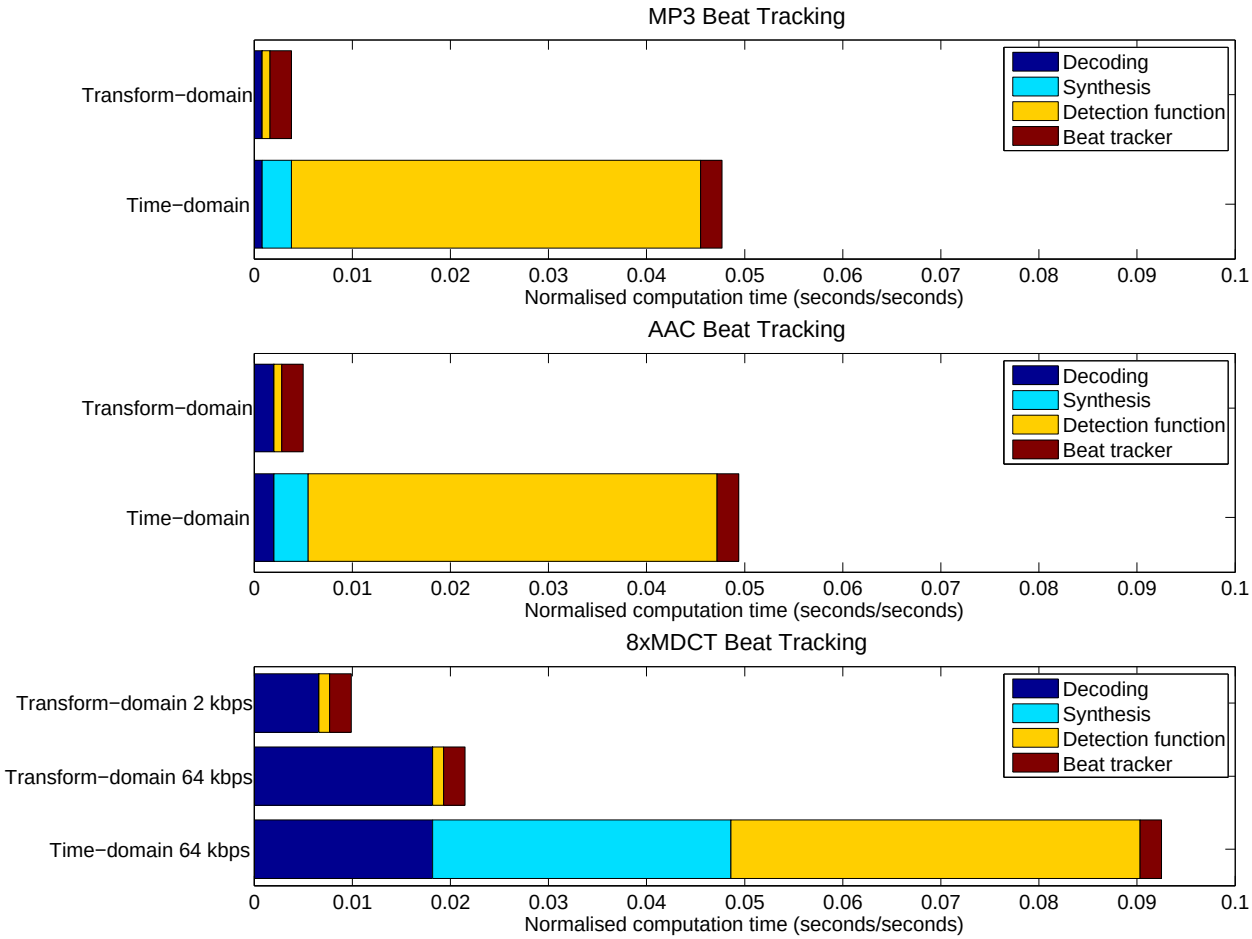

Figure 8.18: *Computation times of the beat tracking systems.*

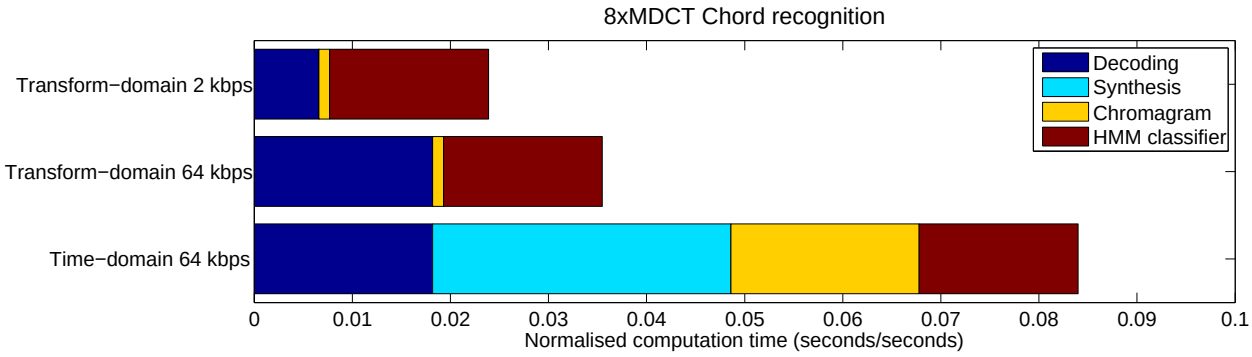

Figure 8.19: *Computation times of the chord recognition systems.*

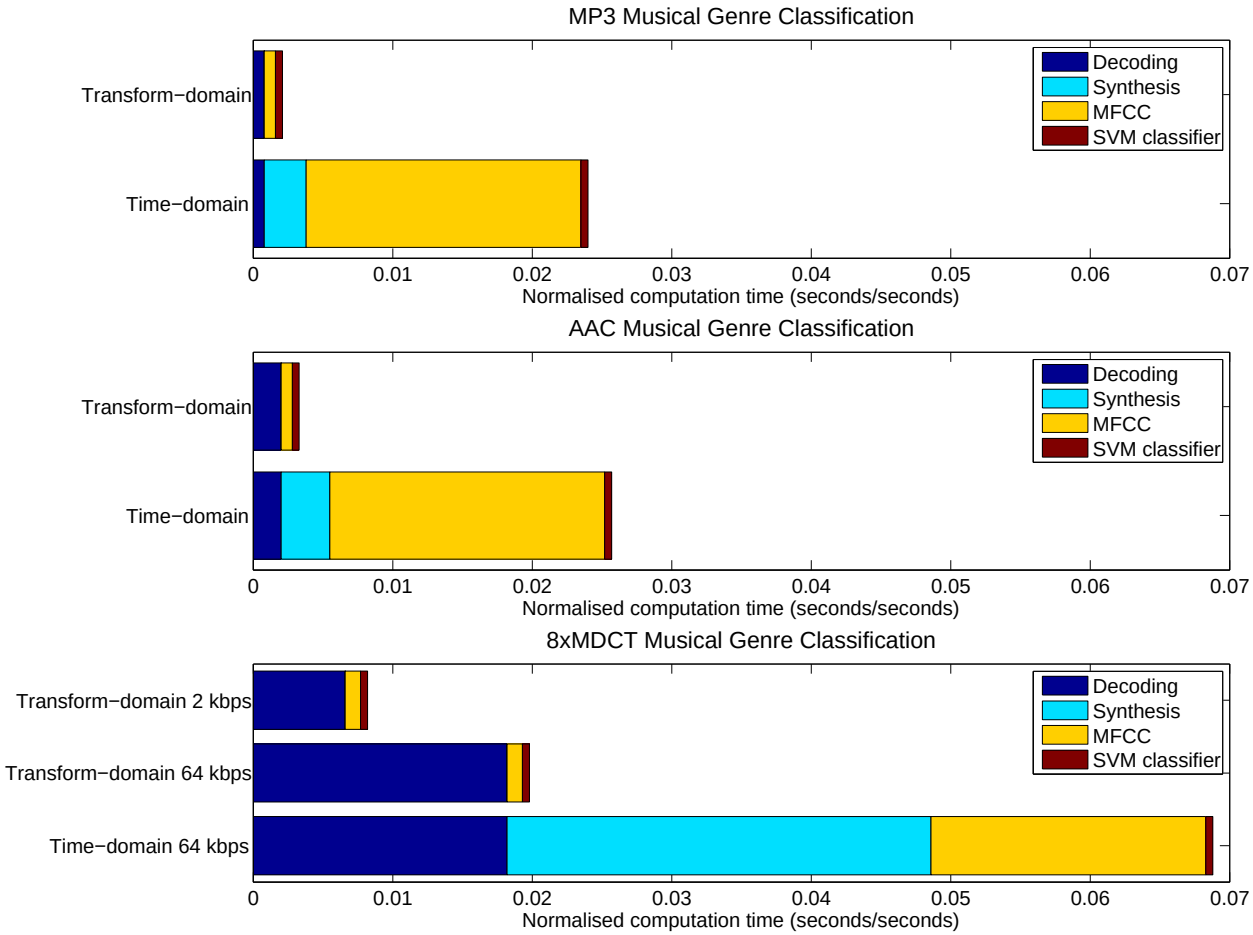

Figure 8.20: *Computation times of the musical genre classification systems.*

8.5 Conclusions

We studied in this chapter transform-domain audio indexing. We have considered 3 audio codecs, namely the standard MP3, the standard AAC, and our proposed codec based on the union of 8 MDCT bases. And we have considered 3 applications, namely beat tracking, chord recognition, and musical genre classification. For each codec, and for each application, we have proposed a fast and efficient transform-domain mid-level representation, except for the MP3/AAC codecs and the chord recogntion. Indeed, only the proposed codec has enough frequency resolution to build an efficient chromagram for chord recognition. On the other hand, the MP3 and the AAC systems have the advantage of very fast implementations as compared to our 8xMDCT codec.

Future work would investigate other audio codecs, such as e.g. MPEG-4 SSC, which is based on a parametric model that may be very useful for audio indexing. It would also be interesting to investigate other audio indexing applications, such as e.g. audio fingerprinting.

Chapter 9

Conclusions

9.1 General conclusion

The idea behind this thesis was to investigate the application of new signal representation approaches to audio coding. These new approaches are based on techniques that come from a research domain known as "sparse signal representations". These techniques allow the design of signal representations that have several advantages over the traditional approaches, including better resolution and flexibility.

However, the main drawback of these methods is their computational complexity. When these techniques have been developed (e.g. Matching Pursuit in 1992), the complexity problem prevent their use in real-world applications. It is only since a few years ago, with the increase in computer performance and especially the development of efficient algorithms (e.g. MPTK), that these techniques have found interests in practical applications. As an example, the system proposed in Chapter 6 obtains computation times around 3 times the real-time, which is not excessive and is acceptable for e.g. off-line sound databases.

Of course, the proposed methods are still slower than the traditional transform coding, but we have shown that our approach have several advantages over transform coding, and consequently the performance/complexity tradeoff tends now to be in our favor. These advantages include better performance than transform coding at low bitrates and similar "transparent-quality" at high bitrates. Our approach thus allows a more "universal" audio coder than the traditional transform coders.

However, the proposed methods are still a first step towards efficient universal audio coding because we have shown that the obtained performance is still highly signal dependant. Particularly, the performance of our approach is high only for monophonic signals. For polyphonic signals, the obtained performance is not as satisfactory, but we have shown that, in that case, our approach has another advantage over transform coding, which is efficient transform-domain audio indexing. We have shown that a combined audio coding and audio indexing system is now possible.

9.2 Future research

We propose below several possible ways for further research.

Signal Representation

The signal model proposed in this thesis is based on a union of MDCT bases. One can imagine other ways to build overcomplete dictionaries. One possibility is to consider the complex extension of the MDCT, known as the MCLT. This approach has been introduced in Chapter 7, but it still needs to be further investigated. A complete audio codec based on MCLT still needs to be implemented and a proper objective and subjective evaluation is required. Another possibility we have not considered is to increase the overlap between the MDCT atoms, which leads to a more shift invariant representation. This approach has been investigated in a recent paper [SM08], they have also considered atoms with asymmetric windows, such as gammatones.

Other possibilities include considering other decomposition algorithms. In this thesis, we use the Matching Pursuit (MP) algorithm (and slightly modified versions of MP) to find sparse approximations in an overcomplete dictionary. However, MP is not the only algorithm to find sparse approximations. Others exist, and they appeared to perform better than MP for some applications (see Chapter 3 for a short review of sparse signal representations). Particularly, we believe that some variants of MP, such as the Gradient Pursuit [BD08] presented in Chapter 3 may potentially significantly increase the performance of our coder. We present in Annex B a preliminary experiment that shows that sparser approximations are obtained with GP. Further studies would then develop and evaluate a coder based on GP.

Another perspective would be to consider structured representations (see e.g. the thesis of P. Leveau [Lev07]). This is a slightly different concept as a signal is here represented as a combination of "musical objects" instead of individual atoms. While the union of MDCT bases is closer to the traditional filter-bank based approach, these structured representations are closer to sinusoidal modeling approaches used in low-bitrates audio codecs such as MPEG-4 SSC. One interesting problem would be to combine these two concepts. It is particularly not clear how to design a single coding paradigm for these different representations.

Audio coding

In the proposed audio codec, we use a relatively simple source coding algorithm. Further studies would investigate better algorithms such as for example bitplane encoding based on context-based arithmetic coding, or other model-based source coding algorithms (see e.g. the thesis of M. Oger [Oge07]). It would also be interesting to consider non-scalable audio coding, where the audio codec is optimized for a particular bitrate.

Also, one of the weak point in the proposed codec and that needs improvements is the psychoacoustic model. As explained previously, we just have extended a transform-coding model to our multiresolution model, this is obviously far from optimal. Further research would thus investigate better psychoacoustic models, we believe that an improved psychoacoustic model would results in significant improvements. Moreover, a question not yet answered is whether it is better to integrate the psychoacoustic model directly in the decomposition algorithm, an approach similar as the psychoacoustic matching Pursuit from Heusdens et al [HVK02]. This would require an implementation of a psychoacoustic model in MPTK which is far from being straightforward, and which may also increase significantly the computational cost.

A possible extension would be to investigate the integer MDCT [GSK01, LRYK07]. This integer transform is useful for lossless audio coding, it is used for example in the standard MPEG-4 SLS [YRXK06]. Using the integer MDCT instead of the standard MDCT in our proposed audio codec would allows a fine-grain scalable audio coder that operates from very low bitrates to lossless.

Another possible future work would investigate bandwith extension and stereo coding. One

could imagine coding pair of atoms instead of single atoms: either pair of low/high frequency atoms, or pair of left/right channel atoms. This is particularly relevant in our case because matching pursuit has a straightforward multichannel implementation. Stereo MP has already been investigated for source separation [Gri02] and musical instrument recognition [SLD07], but not yet for audio coding.

Transform-domain audio indexing

One natural extension to this work would be the investigation of other audio codecs. Firstly, it would be interesting to investigate other standard codecs. As an example, we believe that parametric codecs such as MPEG-4 HILN and MPEG-4 SSC would be very useful for audio indexing applications such as chord recognition or beat tracking. Indeed, these parametric coders perform a sinusoidal analysis which estimates with precision the frequency of sinusoidal waves, this can be useful for computing a chromagram, but also for other frequency-related applications. Moreover, MPEG-4 SSC also estimates a transient component, which can be useful for onset-related applications such as beat tracking. The main drawback of these codecs as compared to transform-based standard codecs is that they are not very widely spread, and even probably not used at all. Secondly, an interesting question is whether the audio codec based on the adaptive dictionary and presented in chapter 6 gives the same results as the simple codec based on the union of 8 MDCT bases. As this "adaptive codec" has several advantages such as increased performance at high bitrates and lower encoding complexity, it would be interesting to study the application of such an approach to transform-domain audio indexing.

Another natural extension would be to consider other audio indexing applications. One interesting example is the problem of finding the nearest neighboor of a given signal in a set of coded signal. This problem is useful for e.g. audio fingerprinting. A recent work [JV08] studied this problem for the case of images, where the sparseness of a representation in a overcomplete dictionary is used to design fast algorithms. It would be interesting to study this problem in the case of audio.

Publications

Publications related to this thesis

- **E. Ravelli, G. Richard, and L. Daudet**. Union of MDCT bases for audio coding. *IEEE Transactions on Audio, Speech and Language Processing (accepted for future publication)*, 2008.
- **E. Ravelli and L. Daudet**. Embedded polar quantization. *IEEE Signal Processing Letters*, 14(10):657-660, Oct. 2007.
- **E. Ravelli, G. Richard, and L. Daudet**. Fast MIR in a sparse transform domain. In *Proc. Of 9th International Conference on Music Information Retrieval (accepted for future publication)*, Sep. 2008.
- **E. Ravelli, G. Richard, and L. Daudet**. Matching pursuit in adaptive dictionaries for scalable audio coding. In *Proc. Of 16th European Signal Processing Conference (accepted for future publication)*, Aug. 2008.
- **E. Ravelli, G. Richard, and L. Daudet**. Extending fine-grain scalable audio coding to very low bitrates using overcomplete dictionaries. In *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA'07)*, pages 195-198, Oct. 2007.
- **E. Ravelli and L. Daudet**. Representations of audio signals in overcomplete dictionaries: What is the link between redundancy factor and coding properties? In *Proc. Of 9th Int. Conf. on Digital Audio Effects (DAFX'06)*, pages 267-270, Sep. 2006.

Other publications

- **E. Ravelli, J. P. Bello, and M. Sandler**. Automatic rhythm modification of drum loops. *IEEE Signal Processing Letters*, 14:228-231, Apr. 2007.
- **Grégory Cornuz, Emmanuel Ravelli, Pierre Leveau and Laurent Daudet**. Object coding of harmonic sounds using sparse and structured representations. *Proc. of 10th Int. Conf. on Digital Audio Effects (DAFX'07)*, pages 41-46, Sep. 2007.
- **E. Ravelli, P. Gournay, and R. Lefebvre**. A two-stage MLP+NLMS lossless coder for stereo audio. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing, (ICASSP'06)*, volume 5, pages 177-180, May 2006.
- **J. P. Bello, E. Ravelli and M. Sandler**. Drum sound analysis for the manipulation of rhythm in drum loops. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing, (ICASSP'06)*, volume 5, pages 233-236, May 2006.

- **E. Ravelli, J. P. Bello, and M. Sandler**. Fast implementation for non-linear time-scaling of stereo signals. In *Proc. Of 8th Int. Conf. on Digital Audio Effects (DAFX'05)*, pages 182-185, Sep. 2005.

Bibliography

[AMDM87] J.-P. Adoul, P. Mabilleau, M. Delprat, and S. Morissette. Fast CELP coding based on algebraic codes. In *Proc. IEEE Int. Conf. Acoustics, Speech and Sig. Proc.*, volume 12, pages 1957–1960, Apr. 1987.

[AS84] B. S. Atal and M. R. Schroeder. Stochastic coding of speech signals at very low bit rates. In *Proc. Int. Conf. Commun.- ICC84*, volume 2, pages 1610–1613, May 1984.

[ASMN+99] O.K. Al-Shaykh, E. Miloslavsky, T. Nomura, R. Neff, and A. Zakhor. Video compression using matching pursuits. *IEEE Trans. Circuits Syst. Video Technol.*, 9(1):123–143, Feb. 1999.

[BCE+06] J. Bergstra, N. Casagrande, D. Erhan, D. Eck, and B. Kégl. Aggregate features and ADABOOST for music classification. *Machine Learning*, 65(2-3):473 – 484, 2006.

[BD08] T. Blumensath and M. E. Davies. Gradient pursuits. *IEEE Trans. on Sig. Proc.*, 56(6):2370–2382, June 2008.

[BDA+05] J.P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M.B. Sandler. A tutorial on onset detection in music signals. *IEEE Trans. Speech and Audio Proc.*, 13(5):1035–1047, Sept. 2005.

[BDDS04] J.P. Bello, C. Duxbury, M. Davies, and M. Sandler. On the use of phase and energy for musical onset detection in the complex domain. *IEEE Sig. Proc. Letters*, 11(6):553–556, 2004.

[BF96] H. Brunk and N. Farvardin. Fixed-rate successively refinable scalar quantizers. In *Proc. IEEE Data Compression Conf.*, pages 250–259, 1996.

[BF98] H. Brunk and N. Farvardin. Embedded trellis coded quantization. In *Proc. IEEE Data Compression Conf.*, pages 93–102, 1998.

[BG03] M. Bosi and R. E. Goldberg. *Introduction to digital audio coding and standards.* Kluwer Academic Publishers, 2003.

[BKS00] K. Brandenburg, O. Kunz, and A. Sugiyama. MPEG-4 natural audio coding. *Signal Processing*, 15:423–444, 2000.

[BP92] J. Brown and M. Puckette. An efficient algorithm for the calculation of a constant Q transform. *J. Acoust. Soc. America*, 92(5):2698–2701, 1992.

[BP05] J. P. Bello and J. Pickens. A robust mid-level representation for harmonic content in music signals. In *Proc. Int. Conf. Music Inf. Retrieval*, pages 304–311, 2005.

[Bro91] J. Brown. Calculation of a constant Q spectral transform. *J. Acoust. Soc. America*, 89(1):425–434, 1991.

[CARKD99] S. F. Cotter, J. Adler, B. Rao, and K. Kreutz-Delgado. Forward sequential algorithms for best basis selection. In *IEEE Proc. Vision Image Sig. Proc.*, pages 235–244, 1999.

[CCW92] R.R. Coifman, R.R. Coifman, and M.V. Wickerhauser. Entropy-based algorithms for best basis selection. *IEEE Trans. Inform. Theory*, 38(2):713–718, 1992.

[CDS99] S. Chen, D. Donoho, and M. Saunders. Atomic decomposition by Basis Pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1999.

[CRLD07] G. Cornuz, E. Ravelli, P. Leveau, and L. Daudet. Object coding of harmonic sounds using sparse and structured representations. In *Proc.10th Int. Conf. on Digital Audio Effects*, pages 41–46, 2007.

[Dau88] I. Daubechies. Time-frequency localization operators: a geometric phase space approach. *IEEE Trans. on Inf. Theory*, 34(4):605–612, Jul. 1988.

[DD06] M. E. Davies and L. Daudet. Sparse audio representation using the MCLT. *Signal Processing*, 86(3):457–470, 2006.

[Dix01] S. Dixon. Automatic extraction of tempo and beat from expressive performances. *J. New Music Research*, 30(1):3958, 2001.

[DKK97] T. Dau, B. Kollmeier, and A. Kohlrausch. Modeling auditory processing of amplitude modulation: I. modulation detection and masking with narrowband carriers. *J. Acoust. Soc. Amer.*, 102:28922905, 1997.

[DLKK02] M. Dietz, L. Liljeryd, K. Kjörling, and O. Kunz. Spectral band replication, a novel approach in audio coding. In *Proc. of the 112th AES Convention*, 2002. Paper 5553.

[DM80] S. Davis and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Trans. Acoust., Speech, Sig. Proc.*, 28(4):357–366, 1980.

[DMA97] G. Davis, S. Mallat, and M. Avellaneda. Greedy adaptive approximation. *J. Constr. Approx.*, 13:57–98, 1997.

[DMP91] P. Duhamel, Y. Mahieux, and J.P. Petit. A fast algorithm for the implementation of filter banks based on 'time domain aliasing cancellation'. In *Proc. IEEE Int. Conf. Acoustics, Speech and Sig. Proc.*, pages 2209–2212, Apr. 1991.

[Don95] D.L. Donoho. De-noising by soft-thresholding. *IEEE Trans. Inform. Theory*, 41(3):613–627, May 1995.

[DP07] M. E. P. Davies and M. D. Plumbley. Context-dependant beat tracking of musical audio. *IEEE Trans. on Audio, Speech and Lang. Proc.*, 15(3):1009–1020, 2007.

[DPK96] T. Dau, D. Pschel, and A. Kohlrausch. A quantitative model of the effective signal processing in the auditory system: I. Model structure. *J. Acoust. Soc. Amer.*, 99:3615–3622, 1996.

[dSO02] A.C. den Brinker, E.G.P. Schuijers, and A.W.J. Oomen. Parametric coding for high-quality audio. In *Proc. of the 112th AES Convention*, 2002. Paper 5554.

[Dun06] C. Dunn. Scalable bitplane runlength coding. In *Proc. 120th Audio Eng. Soc. Conv.*, 2006. paper 6749.

[EA06] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Trans. Image Proc.*, 15(12):3736–3745, Dec. 2006.

[Edl89] B. Edler. Codierung von audiosignalen mit berlappender transformation und adaptiven fensterfunktionen. *Frequenz*, pages 252–256, 1989.

[EHJS99] E. Ekudden, R. Hagen, I. Johansson, and J. Svedberg. The adaptive multi-rate speech coder. In *Proc. IEEE Workshop on Speech Coding*, pages 117–119, Jun. 1999.

[EJHT04] B. Efron, I. Johnstone, T. Hastie, and R. Tibshirani. Least angle regression. *The Annals of Statistics*, 32(2):407499, 2004.

[EMSZ07] M. Elad, B. Matalon, J. Shtok, and M. Zibulevsky. A wide-angle view at iterated shrinkage algorithms. *SPIE, Wavelet XII*, 2007.

[EPF96] B. Edler, H. Purnhagen, and C. Ferekidis. ASAC - analysis/synthesis audio codec for very low bit rates. In *Proc. 110th AES Conv.*, May 1996. Preprint 4179.

[FAA08] FAAC. FAAC and FAAD webpage, 2008. http://sourceforge.net/projects/faac/.

[FBD+96] L. D. Fielder, M. Bosi, G. A. Davidson, M. Davis, C. Todd, and S. Vernon. AC-2 and AC-3: Low Complexity Transform-Based Audio Coding. In *Collected Papers on Digital Audio Bit-Rate Reduction*, pages 54–72, 1996.

[FG06] C. Fevotte and S.J. Godsill. A bayesian approach for blind separation of sparse sources. *IEEE Trans. on Audio, Speech and Lang. Proc.*, 14(6):2174–2188, Nov. 2006.

[FiVVF06] R.M. Figueras i Ventura, P. Vandergheynst, and P. Frossard. Low-rate and flexible image coding with redundant representations. *IEEE Trans. on Image Proc.*, 15(3):726–739, Mar. 2006.

[FNW07] M.A.T. Figueiredo, R.D. Nowak, and S.J. Wright. Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems. *IEEE Journal of Selected Topics in Signal Processing*, 1(4):586–597, Dec. 2007.

[FTDG08] C. Fevotte, B. Torresani, L. Daudet, and S. J. Godsill. Sparse linear regression with structured priors and application to denoising of musical audio. *IEEE Trans. on Audio, Speech and Lang. Proc.*, 16(1):174–185, 2008.

[Fuj99] T. Fujishima. Realtime chord recognition of musical sound: A system using common lisp music. In *Proc. Int. Comput. Music Conf.*, pages 464–467, 1999.

[Gab47] D. Gabor. Acoustical quanta and the theory of hearing. *Nature*, pages 591–516, 1947.

[Gal78] R. G. Gallager. Variations on a theme by huffman. *IEEE Trans. Inform. Theory*, 24(6):668–674, Nov. 1978.

[GBM+96] R. Gribonval, E. Bacry, S. Mallat, P. Depalle, and X. Rodet. Analysis of sound signals with high resolution matching pursuit. In *Proc. of the International Symposium on Time-Frequency and Time-Scale Analysis*, pages 125–128, 1996.

[GG92] A. Gersho and R. M. Gray. *Vector Quantization and Signal Compression*. Kluwer Academic Publishers, 1992.

[GH04] E. Gomez and P. Herrera. Estimating the tonality of polyphonic audio files: Cognitive versus machine learning modelling strategies. In *Proc. of the 5th ISMIR*, pages 92–95, 2004.

[GMPV04] L. Granai, E. Maggio, L. Peotta, and P. Vandergheynst. Hybrid video coding based on bidimensional matching pursuit. *EURASIP Journal on Applied Signal Processing*, 1:2705–2714, 2004.

[GN98] R.M. Gray and D.L. Neuhoff. Quantization. *IEEE Trans. Inform. Theory*, 44(6):2325–2383, Oct. 1998.

[Gol66] S.W. Golomb. Run-length encodings. *IEEE Trans. Inform. Theory*, 12(3):399–401, 1966.

[Goo97] M. Goodwin. Matching pursuit with damped sinusoids. In *Proc. IEEE Int. Conf. Acoustics, Speech and Sig. Proc.*, volume 3, pages 2037–2040 vol.3, 1997.

[Goo01] M.M. Goodwin. Multiscale overlap-add sinusoidal modeling using matching pursuit and refinements. In *Proc. Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 207–210, Oct. 2001.

[Gri99] R. Gribonval. *Approximations non-linéaires pour l'analyse des signaux sonores*. PhD thesis, Université Paris-IX Dauphine, 1999.

[Gri02] R. Gribonval. Sparse decomposition of stereo signals with matching pursuit and application to blind separation of more than two sources from a stereo mixture. In *Proc. IEEE Int. Conf. Acoustics, Speech and Sig. Proc.*, volume 3, pages 3057–3060, 2002.

[GS92] E. B. George and M. J. T. Smith. Analysis-by-synthesis/overlap-add sinusoidal modeling applied to the analysis and synthesis of musical tones. *J. Audio Eng. Soc.*, 40(6):497–516, June 1992.

[GSK01] R. Geiger, T. Sporer, and J. Koller. Audio coding based on integer transforms. In *Proc. 111th AES Convention*, 2001. preprint 5471.

[Hai04] S. Hainsworth. *Techniques for the Automated Analysis of Musical Audio*. PhD thesis, Dept. Eng., Cambridge University, 2004.

[HK00] M. Hansen and B. Kollmeier. Objective modelling of speech quality with a psychoacoustically validated auditory model. *J. Audio Eng. Soc.*, 48:395–409, 2000.

[HK06] R. Hubert and B. Kollmeier. PEMO-Q–A new method for objective audio quality assessment using a model of auditory perception. *IEEE Trans. on Audio, Speech and Lang. Proc.*, 14(6):1902–1911, Nov. 2006.

[HS05] C. Harte and M. Sandler. Automatic chord identification using a quantized chromagram. In *Proceedings of the 118th AES Convention*, May 2005.

[HS08] A. Holzapfel and Y. Stylianou. Musical genre classification using nonnegative matrix factorization-based features. *IEEE Trans. on Audio, Speech and Lang. Proc.*, 16(2):424–434, 2008.

[Huf52] D. A. Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the Institute of Radio Engineers*, 40(9):1098–1101, 1952.

[HVK02] R. Heusdens, R. Vafin, and W. B. Kleijn. Sinusoidal modeling using psychoacoustic-adaptive matching pursuits. *IEEE Sig. Proc. Letters*, 9(8):262–265, Aug. 2002.

[HXDC07] Z. He, S. Xie, S. Ding, and A. Cichocki. Convolutive blind source separation in the frequency domain based on sparse representation. *IEEE Trans. on Audio, Speech and Lang. Proc.*, 15(5):1551–1563, July 2007.

[IMM95] N. Iwakami, T. Moriya, and S. Miki. High-quality audio-coding at less than 64 kbit/s by using transform-domain weighted interleave vector quantization (TwinVQ). In *Proc. IEEE Int. Conf. Acoustics, Speech and Sig. Proc.*, volume 5, pages 3095–3098, May 1995.

[ISO92] ISO/IEC, JTC1/SC29/WG11 MPEG. Information technology - coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s - part 3: Audio, IS11172-3 1992.

[ISO96] ISO/IEC, JTC1/SC29/WG11 MPEG. Report on the formal subjective listening tests of MPEG-2 NBC multichannel audio coding, Nov. 1996. Tech. report N1419.

[ISO98a] ISO/IEC, JTC1/SC29/WG11 MPEG. Report on the MPEG-2 AAC stereo verification tests, Feb. 1998. Tech. report N2006.

[ISO98b] ISO/IEC, JTC1/SC29/WG11 MPEG. Information technology - generic coding of moving pictures and associated audio information - part 3: Audio, IS13818-3 1998.

[ISO99] ISO/IEC, JTC1/SC29/WG11 MPEG. Report on the mpeg-4 audio version 2 verification test, Dec. 1999. Tech. report N3075.

[ISO01] ISO/IEC, JTC1/SC29/WG11 MPEG. Information technology - coding of audio-visual objects - part 3: Audio, IS14496-3 2001.

[ISO03] ISO/IEC, JTC1/SC29/WG11 MPEG. Report on informal MPEG-4 extension 1 (bandwidth extension) verification tests, March 2003. Tech. report N5571.

[ISO04a] ISO/IEC. JPEG 2000 image coding system: Core coding system, IS15444-1 2004.

[ISO04b] ISO/IEC, JTC1/SC29/WG11 MPEG. Report on the verification tests of MPEG-4 parametric coding for high quality audio, July 2004. Tech. report N6675.

[ITU97] ITU-R. Recommendation BS.1116-1. Methods for the subjective assessment of small impairments in audio systems including multichannel sound systems, 1997.

[ITU98] ITU-R. Recommendation BS.1387. Method for Objective Measurements of Perceived Audio Quality, 1998.

[ITU01] ITU-R. Draft Revision to BS.1387. Method for Objective Measurements of Perceived Audio Quality, Document 6/BL/30-E, July 2001.

[ITU03] ITU-R. Recommendation BS.1534-1. Method for the subjective assessment of intermediate quality levels of coding systems, 2003.

[iTu08] iTunes. Apple iTunes 7 webpage, 2008. http://www.apple.com/fr/itunes/download/.

[JCMW95] S. Jaggi, W.C. Carl, S. Mallat, and A.S. Willsky. High resolution pursuit for feature extraction. Technical report, MIT, Nov. 1995.

[Joh88] J. D. Johnston. Transform coding of audio signals using perceptual noise criteria. *IEEE Journal on selected areas in communications*, 6(2):314–323, Feb. 1988.

[JV08] P. Jost and P. Vandergheynst. On finding nearest neighbors in a set of compressible signals. *To appear in IEEE Trans. on Sig. Proc.*, 2008.

[JVF06] P. Jost, P. Vandergheynst, and P. Frossard. Tree-based pursuit: Algorithm and properties. *IEEE Trans. on Sig. Proc.*, 54(12):4685–4697, Dec. 2006.

[Kab03] P. Kabal. An examination and interpretation of ITU-R BS.1387: Perceptual evaluation of audio quality. Technical report, McGill, 2003.

[KEA06] A. P. Klapuri, A. J. Eronen, and J. T. Astola. Analysis of the meter of acoustic musical signals. *IEEE Trans. on Audio, Speech and Lang. Proc.*, 14(1):342–355, 2006.

[KJH04] P. Korten, J. Jensen, and R. Heusdens. High rate spherical quantization of sinusoidal parameters. In *Proc. EUSIPCO*, pages 1757–1760, 2004.

[KKL+07] S. J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. A method for large-scale l1-regularized least squares. *IEEE Journal on Selected Topics in Signal Processing*, 4(1):606–617, 2007.

[KLK02] J. L. Kim, K. L., and T. Kim. Adaptive reconstruction for embedded quantisation. *Electronics Letters*, 38(18):1065–1067, Aug. 2002.

[KQG06] S. Kiranyaz, Ahmad Farooq Qureshi, and M. Gabbouj. A generic audio classification and segmentation approach for multimedia indexing and retrieval. *IEEE Trans. on Audio, Speech and Lang. Proc.*, 14(3):1062–1081, 2006.

[Krs06] S. Gribonval R. Krstulovic. MPTK: Matching pursuit made tractable. In *Proc. Int. Conf. on Acoustics, Speech, and Sig. Proc.*, volume 3, pages 496–499, 2006.

[LAM08] LAME. LAME mp3 encoder webpage, 2008. http://lame.sourceforge.net.

[Lan83] G. G. Langdon. An adaptive run-length encoding algorithm. *IBM Tech. Discl. Bull.*, 26:3783–3785, 1983.

[LASM90] C. Laflamme, J.-P. Adoul, H.Y. Su, and S. Morissette. On reducing computational complexity of codebook search in CELP coder through the use of algebraic codes. In *Proc. IEEE Int. Conf. Acoustics, Speech and Sig. Proc.*, pages 177–180, Apr. 1990.

[Lev07] P. Leveau. *Décompositions parcimonieuses structurées: Application à la représentation objet de la musique.* PhD thesis, Université Pierre et Marie Curie, 2007.

[Li02] J. Li. Embedded audio coding (EAC) with implicit auditory masking. In *Proc. ACM Multimedia*, pages 592–601, 2002.

[lib08] libMAD. libMAD mpeg audio decoder webpage, 2008. http://www.underbit.com/products/mad/.

[LLV+08] P. Leveau, P. Leveau, E. Vincent, G. Richard, and L. Daudet. Instrument-specific harmonic atoms for mid-level music representation. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(1):116–128, 2008.

[LRYK07] T. Li, S. Rahardja, R. Yu, and S. N. Koh. On Integer MDCT for Perceptual Audio Coding. *IEEE Trans. on Audio, Speech and Lang. Proc.*, 15(8):2236–2248, Nov. 2007.

[LS98] S. Levine and J. O. Smith. A sines+transient+noise audio representation for data compression and time/pitch scale modifications. In *Proc. of 115th AES Conv.*, Sept. 1998.

[LS08] K. Lee and M. Slaney. Acoustic chord transcription and key extraction from audio using key-dependent HMMs trained on synthesized audio. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):291–301, 2008.

[LSLA94] R. Lefebvre, R. Salami, C. Laflamme, and J.-P. Adoul. High quality coding of wideband audio signals using transform coded excitation (TCX). In *Proc. IEEE Int. Conf. Acoustics, Speech and Sig. Proc.*, volume 1, pages 193–196, Apr. 1994.

[LVS97] S.N. Levine, T.S. Verma, and J.O. Smith. Alias-free, multiresolution sinusoidal modeling for polyphonic, wideband audio. In *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, Oct. 1997.

[Mal90] H. S. Malvar. Lapped transforms for efficient transform/subband coding. *IEEE Trans. on Acoustics, Speech, and Sig. Proc.*, 38(6):969–978, June 1990.

[Mal98] S. Mallat. *A wavelet tour of signal processing.* Academic Press, 1998.

[Mal99a] H. S. Malvar. Fast progressive wavelet coding. In *Proc. Data Compression Conference*, pages 336–343, 1999.

[Mal99b] H. S. Malvar. A modulated complex lapped transform and its applications to audio processing. In *Proc. IEEE Int. Conf. Acoustics, Speech and Sig. Proc.*, volume 3, pages 1421–1424 vol.3, 1999.

[Max60] J. Max. Quantizing for minimum distortion. *IRE Trans. Inf. Thy.*, 6(1):7–12, Mar. 1960.

[MBB+05] J. Makinen, B. Bessette, S. Bruhn, P. Ojala, R. Salami, and A. Taleb. AMR-WB+: a new audio coding standard for 3rd generation mobile audio services. In *Proc. Int. Conf. on Acoustics, Speech, and Sig. Proc.*, volume 2, pages 1109–1112, March 2005.

[MD03] S. Merdjani and L. Daudet. Direct estimation of frequency from MDCT-encoded files. In *Proc. of DAFX 03*, 2003.

[MDZ94] S. Mallat, G. Davis, and Z. Zhang. Adaptive time-frequency decompositions. *SPIE J. Opt. Eng.*, 33:2183–2191, 1994.

[MJV+07] G.. Monaci, P.. Jost, P.. Vandergheynst, B.. Mailhe, S.. Lesage, and R. Gribonval. Learning multimodal dictionaries. *IEEE Trans. Image Proc.*, 16(9):2272–2283, Sept. 2007.

[MMES08] J. Mairal, J. Mairal, M. Elad, and G. Sapiro. Sparse representation for color image restoration. *IEEE Trans. Image Proc.*, 17(1):53–69, 2008.

[MQ86] R. McAulay and T. Quatieri. Speech analysis/synthesis based on a sinusoidal representation. *IEEE Trans. on Acoustics, Speech and Sig. Proc.*, 34(4):744–754, Aug. 1986.

[MZ93] S.G. Mallat and Z. Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Trans. on Signal Processing*, 41(12):3397–3415, Dec. 1993.

[Nat95] B. K. Natarajan. Sparse approximate solutions to linear systems. *SIAM J. Comput.*, 24(2):227–234, 1995.

[Nia06] O. A. Niamut. *Rate-distortion optimal time-frequency decompositions for MDCT-based audio coding*. PhD thesis, Technische Universiteit Delft, 2006.

[NLS+99] Y. Nakajima, Y. Lu, M. Sugano, A. Yoneyama, H. Yamagihara, and A. Kurematsu. A fast audio classification from MPEG coded data. In *Proc. IEEE Int. Conf. Acoustics, Speech and Sig. Proc.*, volume 6, pages 3005–3008 vol.6, 1999.

[Nus81] H. Nussbaumer. Pseudo QMF filter bank. *IBM Tech. Discl. Bull.*, 24:3081–3087, Nov. 1981.

[NZ97] R. Neff and A. Zakhor. Very low bit-rate video coding based on matching pursuits. *IEEE Trans. Circuits Syst. Video Technol.*, 7(1):158–171, Feb. 1997.

[Od99] A. Oomen and A. den Brinker. Sinusoids plus noise modeling for audio signals. In *Proc. of the AES 17th Intl. Conf.*, pages 226–232, Sept. 1999.

[Oge07] M. Oger. *Model-based techniques for flexible speech and audio coding*. PhD thesis, Université de Nice-Sophia Antipolis, 2007.

[Opt08] Opticom. Opticom peaq webpage, 2008. http://www.opticom.de/licensing/peaq.html.

[OWS98] E. Ordentlich, M. Weinberger, and G. Seroussi. A low-complexity modeling approach for embedded coding of wavelet coefficients. In *Proc. Data Compression Conference*, pages 408–417, March 1998.

[PA04] F. Pachet and J. J. Aucouturier. Improving timbre similarity: How high is the sky? *J. Negative Results Speech Audio Sci.*, 1(1), 2004.

[Pam06] E. Pampalk. MA toolbox, 2006. http://www.ofai.at/ elias.pampalk/ma.

[Pas76] R. Pasco. *Source coding algorithms for fast data compression*. PhD thesis, Stanford university, 1976.

[Pau04] S. Pauws. Musical key extraction from audio. In *Proc. of the 5th ISMIR*, 2004.

[PB86] J. P. Princen and A. B. Bradley. Analysis/synthesis filter bank design based on time domain aliasing cancellation. *IEEE Trans. Acoust., Speech, Sig. Proc.*, 34(5):1153–1161, Oct. 1986.

[Pea79] W. A. Pearlman. Polar quantization of a complex gaussian random variable. *IEEE Trans. Commun.*, 27:892–899, Jun. 1979.

[PEF98] H. Purnhagen, B. Edler, and C. Ferekidis. Object-based analysis/synthesis audio coder at very low bit rates. In *104th Audio Eng. Soc. Conv.*, 1998. Paper 4747.

[PGV06] L. Peotta, L. Granai, and P. Vandergheynst. Image compression using an edge adapted redundant dictionary and wavelets. *Signal Processing*, 86(3):444–456, 2006.

[PJB87] J. Princen, A. Johnson, and A. Bradley. Subband/transform coding using filter bank designs based on time domain aliasing cancellation. In *Proc. IEEE Int. Conf. Acoustics, Speech and Sig. Proc.*, volume 12, pages 2161–2164, Apr 1987.

[PKS97] S.-H. Park, Y.-B. Kim, and Y.-S. Seo. Multi-layer bit-sliced bit rate scalable audio coding. In *Proceedings of the 103rd AES Convention*, 1997. preprint 4520.

[PM00] H. Purnhagen and N. Meine. HILN-the MPEG-4 parametric audio coding tools. In *Proceedings of the IEEE International Symposium on Circuits and Systems*, volume 3, pages 201–204, May 2000.

[PRK93] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad. Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition. In *Conference Record of the 27th Annual Asilomar Conference on Signals, Systems and Computers*, volume 1, pages 40–44, 1993.

[PS96] N.V. Patel and I.K. Sethi. Audio characterization for video indexing. In *Proc. SPIE*, volume 2670, pages 373–384, 1996.

[PS00] T. Painter and A. Spanias. Perceptual coding of digital audio. *Proceedings of the IEEE*, 88(4):451–515, 2000.

[PV01] S. Pfeiffer and T. Vincent. Formalisation of MPEG- 1 compressed domain audio features. Technical report, Technical Report 01 / 196, CSIRO Mathematical and Information Sciences, Australia, 2001.

[RD06] E. Ravelli and L. Daudet. Representations of audio signals in overcomplete dictionaries: What is the link between redundancy factor and coding properties? In *Proc. Of 9th Int. Conf. on Digital Audio Effects (DAFX'06)*, pages 267–270, Sep. 2006.

[RD07] E. Ravelli and L. Daudet. Embedded polar quantization. *IEEE Signal Processing Letters*, 14(10):657–660, Oct. 2007.

[Ris76] J. Rissanen. Generalized kraft inequality and arithmetic coding. *IBM J. Res. Develop.*, 20:198–203, 1976.

[RKD99] B. D. Rao and K. Kreutz-Delgado. An affine scaling methodology for best basis selection. *IEEE Trans. Sig. Proc.*, 47:187–200, 1999.

[RKT+07] S. Ragot, B. Kovesi, R. Trilling, D. Virette, N. Duc, D. Massaloux, S. Proust, B. Geiser, M. Gartner, S. Schandl, H. Taddei, Yang Gao, E. Shlomot, H. Ehara, K. Yoshida, T. Vaillancourt, R. Salami, Mi Suk Lee, and Do Young Kim. ITU-T G.729.1: An 8-32 Kbit/s Scalable Coder Interoperable with G.729 for Wideband Telephony and Voice Over IP. In *Proc. IEEE Int. Conf. Acoustics, Speech and Sig. Proc.*, volume 4, pages 529–532, Apr. 2007.

[RMB02] M. Raad, A. Mertins, and I. Burnett. Audio coding based on the modulated lapped transform (MLT) and set partitioning in heirarchical trees. In *Proc. 6th World Multiconference on Systemics, Cybernetics and Informatics*, volume 3, pages 303–306, July 2002.

[RMB03] M. Raad, A. Mertins, and I. Burnett. Scalable to lossless audio compression based on perceptual set partitioning in hierarchical trees (PSPIHT). In *Proc. IEEE Int. Conf. Acoustics, Speech and Sig. Proc.*, volume 5, pages 624–627, May 2003.

[Rot83] J. Rothweiler. Polyphase quadrature filters–A new subband coding technique. In *Proc. IEEE Int. Conf. Acoustics, Speech and Sig. Proc.*, volume 8, pages 1280–1283, Apr. 1983.

[RRD07] E. Ravelli, G. Richard, and L. Daudet. Extending fine-grain scalable audio coding to very low bitrates using overcomplete dictionaries. In *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA'07)*, pages 195–198, Oct. 2007.

[RRD08a] E. Ravelli, G. Richard, and L. Daudet. Fast MIR in a sparse transform domain. In *Proc. Of 9th International Conference on Music Information Retrieval (accepted for future publication)*, Sep. 2008.

[RRD08b] E. Ravelli, G. Richard, and L. Daudet. Matching pursuit in adaptive dictionaries for scalable audio coding. In *Proc. Of 16th European Signal Processing Conference (accepted for future publication)*, Aug. 2008.

[RRD08c] E. Ravelli, G. Richard, and L. Daudet. Union of MDCT bases for audio coding. *IEEE Transactions on Audio, Speech and Language Processing (accepted for future publication)*, 2008.

[RRE+03] B.D. Rao, B.D. Rao, K. Engan, S.F. Cotter, J. Palmer, and K. Kreutz-Delgado. Subset selection in noise based on diversity measure minimization. *IEEE Trans. on Sig. Proc.*, 51(3):760–770, 2003.

[SA85] M. Schroeder and B. Atal. Code-excited linear prediction (CELP): High-quality speech at very low bit rates. In *Proc. IEEE Int. Conf. Acoustics, Speech and Sig. Proc.*, volume 10, pages 937–940, 1985.

[Sai04] A. Said. Introduction to arithmetic coding: Theory and practice. Technical report, Hewlett-Packard Laboratories Report HPL-2004-76, 2004.

[SBT00] S. Sardy, A. G. Bruce, and P. Tseng. Block coordinate relaxation methods for nonparametric wavelet denoising. *Journal of Computational and Graphical Statistics*, 9:361–379, 2000.

[Sch98] E. D. Scheirer. Tempo and beat analysis of acoustic musical signals. *J. Acoust. Soc. Am.*, 103(1):588–601, 1998.

[SE03] A. Sheh and D. P. W. Ellis. Chord segmentation and recognition using EM-trained hidden Markov models. In *Proc. Int. Conf. Music Inf. Retrieval*, pages 185–191, 2003.

[SED05] J.-L. Starck, M. Elad, and D.L. Donoho. Image decomposition via the combination of sparse representations and a variational approach. *IEEE Trans. Image Proc.*, 14(10):1570–1582, Oct. 2005.

[Ser89] X. Serra. *A System For Sound Analysis Transformation Synthesis Based on a Deterministic Plus Stochastic Decomposition*. PhD thesis, Stanford University, 1989.

[Sha93] J. M. Shapiro. Embedded image coding using zerotrees of wavelet coefficients. *IEEE Trans. Sig. Proc.*, 41(12):3445–3462, Dec. 1993.

[SK86] P. F. Swaszek and T. W. Ku. Asymptotic performance of unrestricted polar quantizers. *IEEE Trans. Inf. Theory*, 32:330–333, Mar. 1986.

[SL+97] R. Salami, , C. Laflamme, B. Bessette, J.-P. Adoul, K. Jarvinen, J. Vainio, P. Kapaenen, T. Honkanen, and P. Haavisto. Description of GSM enhanced full rate speech codec. In *Proc. IEEE Int. Conf. on Communications (ICC 97)*, volume 2, pages 725–729, 1997.

[SLA+98] R. Salami, C. Laflamme, J.-P. Adoul, A. Kataoka, S. Hayashi, T. Moriya, C. Lamblin, D. Massaloux, S. Proust, P. Kroon, and Y. Shoham. Design and description of CS-ACELP: a toll quality 8 kb/s speech coder. *IEEE Trans. on Speech and Audio Proc.*, 6(2):116–130, Mar. 1998.

[SLD07] David Sodoyer, Pierre Leveau, and Laurent Daudet. Using stereo information for instrument identification in polyphonic mixtures. In *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 259–262, 2007.

[SM08] S. Strahl and A. Mertins. Sparse gammatone signal model optimized for english speech does not match the human auditory filters. *Brain Research (accepted for publication)*, 2008.

[SOdB03] E. Schuijers, W. Oomen, B. den Brinker, and J. Breebaart. Advances in parametric coding for high-quality audio. In *Proceedings of the 114th AES Convention*, 2003. Paper 5852.

[SP96] A. Said and W. A. Pearlman. A new fast and efficient image codec based on set partitioning in hierarchical trees. *IEEE Trans. Circuits Systems Video Tech.*, 6(3):243–250, Jun. 1996.

[SS87] J. O. Smith and X. Serra. PARSHL: A program for the analysis/synthesis of inharmonic sounds based on a sinusoidal representation. In *Proc. International Computer Music Conference*, 1987.

[SSDR08] B.L. Sturm, J.J. Shynk, L. Daudet, and C. Roads. Dark energy in sparse atomic estimations. *IEEE Trans. on Audio, Speech and Lang. Proc.*, 16(3):671–676, 2008.

[SXWK04] X. Shao, C. Xu, Y. Wang, and M.S. Kankanhalli. Automatic music summarization in compressed domain. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '04)*, volume 4, pages iv–261–iv–264, 17–21 May 2004.

[SZM05] S. Strahl, H. Zhou, and A. Mertins. An adaptive tree-based progressive audio compression scheme. In *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 219–222, 16-19 Oct. 2005.

[TC00] G. Tzanetakis and F. Cook. Sound analysis using mpeg compressed audio. In *Proc. IEEE Int. Conf. Acoustics, Speech and Sig. Proc.*, volume 2, pages II761–II764 vol.2, 2000.

[TC02] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Trans. Acoust., Speech, Sig. Proc.*, 10(5):293–302, 2002.

[Tem00] V. Temlyakov. Weak greedy algorithms. *Advances in Computational Mathematics*, 12:213–227, 2000.

[Tib96] R. Tibshirani. Regression shrinkage and selection via the LASSO. *J. Royal. Statist. Soc B.*, 58(1):267–288, 1996.

[TTB+00] T. Thiede, W. C. Treurniet, R. Bitto, C. Schmidmer, T. Sporer, J. G. Beerends, C. Colomes, M. Keyhl, G. Stoll, K. Brandenburg, , and B. Feiten. PEAQ the ITU standard for objective measurement of perceived audio quality. *J. Audio Eng. Soc.*, 48:3–29, 2000.

[Tzo86] K.-H. Tzou. Embedded max quantization. In *Proc. IEEE Int. Conf. Acoustics, Speech and Sig. Proc.*, volume 11, pages 505–508, 1986.

[Vaf04] R. Vafin. *Towards Flexible Audio Coding.* PhD thesis, KTH (Royal Institute of Technology), 2004.

[Vin08] E. Vincent. MUSHRAM: A matlab interface for mushra listening tests, 2008. http://www.elec.qmul.ac.uk/digitalmusic/downloads.

[Vit87] J. S. Vitter. Design and analysis of dynamic huffman codes. *J. ACM*, 34(4):825–845, Oct. 1987.

[VK05] R. Vafin and W.B. Kleijn. Entropy-constrained polar quantization and its application to audio coding. *IEEE Trans. on Audio, Speech and Lang. Proc.*, 13(2):220–232, 2005.

[VK06] R. Vafin and W.B. Kleijn. Rate-distortion optimized quantization in multistage audio coding. *IEEE Trans. on Audio, Speech and Lang. Proc.*, 14(1):311–320, 2006.

[VM99] T.S. Verma and T.H.Y. Meng. Sinusoidal modeling using frame-based perceptually weighted matching pursuits. In *Proc. IEEE Int. Conf. Acoustics, Speech and Sig. Proc.*, volume 2, pages 981–984, Mar. 1999.

[VM00] T.S. Verma and T.H.Y. Meng. A 6kbps to 85kbps scalable audio coder. In *Proc. IEEE Int. Conf. Acoustics, Speech and Sig. Proc.*, volume 2, pages 877–880, Jun. 2000.

[VP07] E. Vincent and M. D. Plumbley. Low bit-rate object coding of musical audio using bayesian harmonic models. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(4):1273–1282, May 2007.

[VPK05] R. Vafin, D. Prakash, and W.B. Kleijn. On frequency quantization in sinusoidal audio coding. *IEEE Sig. Proc. Letters*, 12(3):210–213, 2005.

[vv05] N.H. van Schijndel and S. van de Par. Rate-distortion optimized hybrid sound coding. In *Proc. Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 235–238, 2005.

[Wil80] S. G. Wilson. Magnitude/phase quantization of independent gaussian variates. *IEEE Trans. Commun.*, 28:1924–1929, Nov. 1980.

[WKHP03] M. Wolters, K. Kjorling, D. Homm, and H. Purnhagen. A closer look into MPEG-4 High Efficiency AAC. In *Proceedings of the 115th AES Convention*, 2003. paper 5871.

[WNC87] I. H. Witten, R. M. Neal, and J. G. Cleary. Arithmetic coding for data compression. *Communications of the ACM*, 30:520–540, 1987.

[WV01] Y. Wang and M. Vilermo. A compressed domain beat detector using mp3 audio bitstreams. In *ACM Multimedia*, pages 194–202, 2001.

[YM08] B. J. Yoon and H. S. Malvar. Coding overcomplete representations of audio using the mclt. In *Proc. of IEEE Data Compression Conference*, 2008.

[YRXK06] R. Yu, S. Rahardja, L. Xiao, and C. C. Ko. A fine granular scalable to lossless audio coder. *IEEE Trans. in Audio, Speech, and Language Proc.*, 14(4):1352–1363, July 2006.

[YZ97] L. Yapp and G. Zick. Speech recognition on mpeg/audio encoded files. In *Proc. IEEE International Conference on Multimedia Computing and Systems '97*, pages 624–625, 1997.

[ZEEL02] T. Ziegler, A. Ehret, P. Ekstrand, and M. Lutzky. Enhancing mp3 with SBR: Features and capabilities of the new mp3PRO algorithm. In *Proc. 112th Audio Eng. Soc. Conv.*, 2002. Paper 5560.

[ZW08] J. Zhu and Y. Wang. Complexity-scalable beat detection with MP3 audio bitstreams. *Computer Music Journal*, 32(1):71–87, 2008.

Appendix A

Testing material

We describe here the testing material we use in the chapters 4, 5 and 6 of this thesis. We have chosen five signals, from the dataset of [ISO03] (see Table A.1). This subset was chosen in order to keep critical and rather varied material (from impulsive monoinstrumental signals to complex polyphony). The length of each signal is approximately 10 seconds. We resampled the signals from 48 khz to 44.1 khz and kept only the left channel. Signals and spectrograms are in Figure A.1.

Table A.1: Testing material

Test Item	Description
harp	Harpsichord
bagp	Bagpipes
gloc	Glockenspiel
orch	Orchestral piece
popm	Contemporary pop music

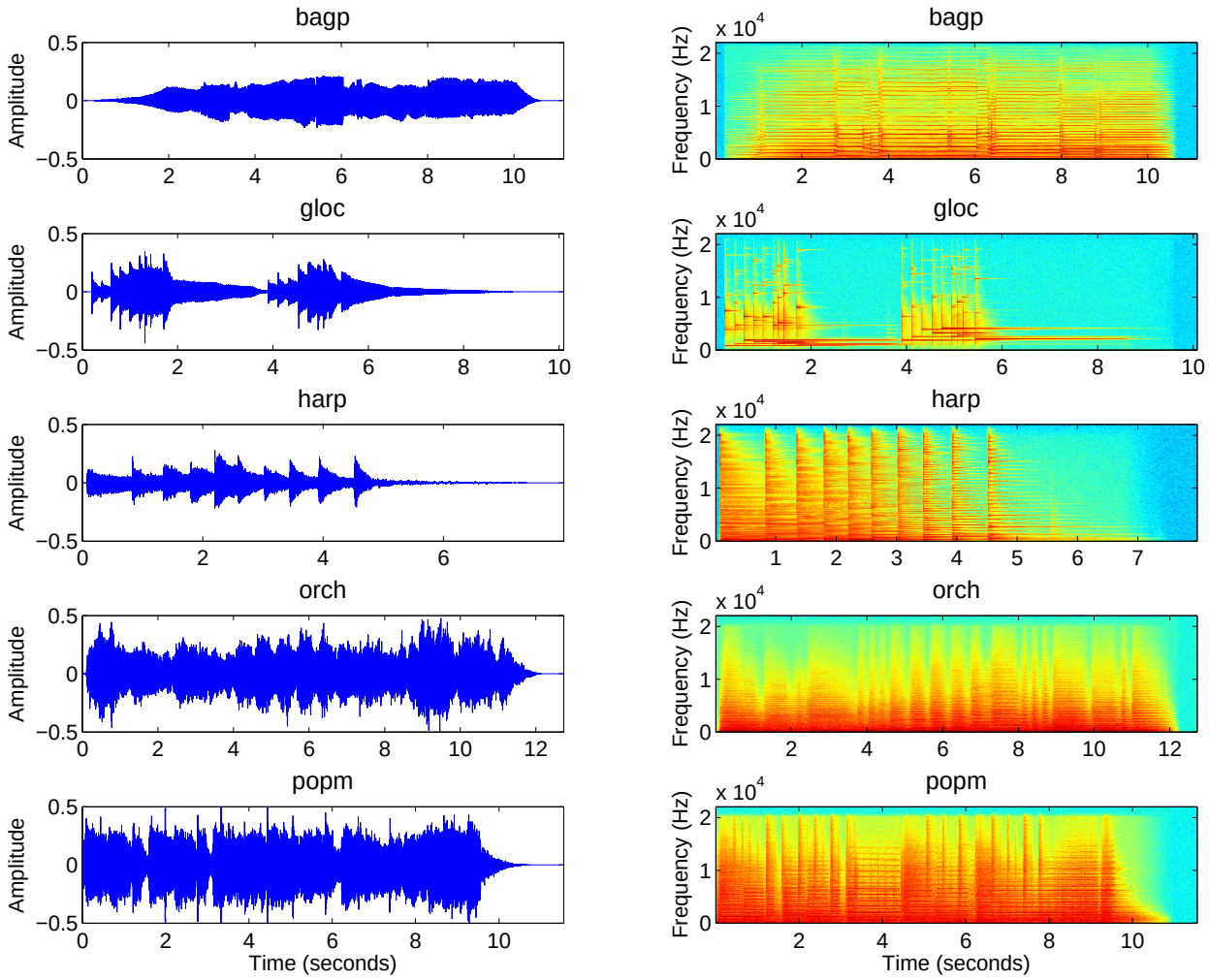

Figure A.1: *Signal and spectrogram of the testing material.*

Appendix B

Gradient pursuit over a union of MDCT bases

We have used in this thesis Matching Pursuit or slightly modified version of Matching Pursuit only. We describe here a preliminary experiment that show the potential benefit of using other greedy algorithms such as Gradient Pursuit [BD08]. We compare in the following three greedy algortihms, the simple Matching Pursuit, Orthogonal Matching Pursuit, and Gradient Pursuit. We first described the experimental setup and then present the results.

Test sounds The algorithms are tested on two sounds. To limit complexity, we use short extracts of approximately two seconds. The first sound is an extract of the glockenspiel signal, caracterized by sharp attacks and long stationary parts. The second sound is an extract of the orchestra signal, caracterized by a rich frequency content.

Dictionary We use a union of 8 MDCT bases with sine-based windows of the following sizes (in samples): 128; 256, 512, 1024, 2048, 4096, 8192, 16384.

Algorithms We compare in this experiment three greedy algorithms: Matching Pursuit (MP) [MDZ94], Orthogonal Matching Pursuit (OMP) using the QR factorization [CARKD99], and Gradient Pursuit (GP) [BD08]. We use a Matlab implementation of these algorithms, called Sparsify and developped by Thomas Blumensath [1]. It is important to remark that the dictionary matrix Φ is not implemented explicitly, due to the memory limitations. Instead, we implement the matrix multiplications $\Phi^T\mathbf{f}$ (with $\mathbf{f}$ a column vector of length N) and $\Phi\mathbf{f}$ (with $\mathbf{f}$ a column vector of length K) using the fast transform described in 4.3.

Machine We use a laptop based on a Core 2 Duo 2GHz processor with 2GB RAM. The algorithms are run on Matlab R2008a (Matlab 7.6.0).

Results The performance of the algorithms are compared using the SNR in function of the number of selected atoms i.e. the number of non-zero coefficients i.e. the 0-like-norm of the approximation. Results are on Fig. B.1. As a reference, we also give the SNR obtained by keeping the first most correlated atoms in an orthogonal dictionary (MDCT with window size 2048 samples).

[1] The Matlab Sparsify Toolbox is released under a GNU/GPL license and is available at the following adress: http://www.see.ed.ac.uk/ tblumens/sparsify/sparsify.html

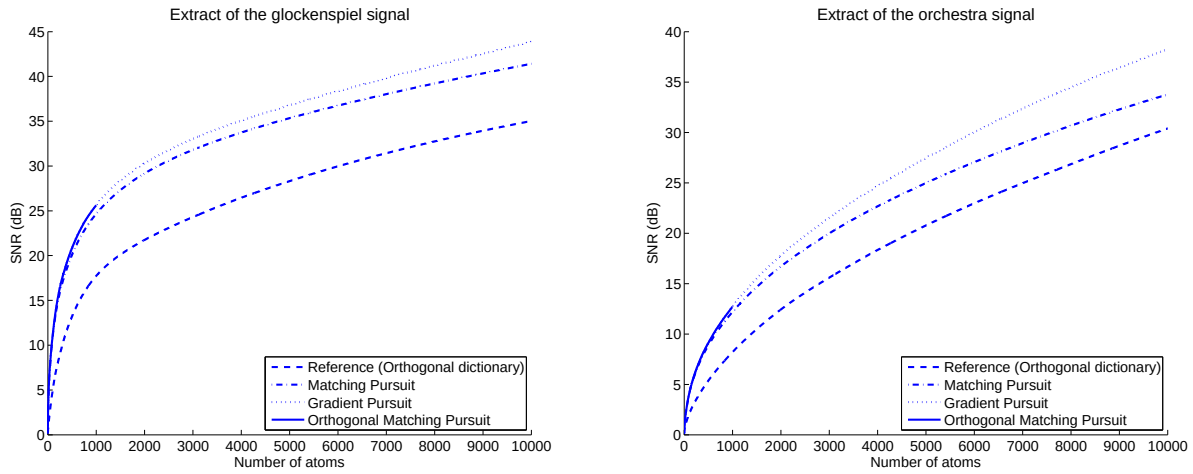

Figure B.1: *Performance comparison of several greedy algorithms.*

It is important to remark that due to the huge memory requirement of OMP, the algorithm has been stopped after 1000 iterations, this explain the limited curve of OMP. We also remark that the performance of OMP and GP are nearly the same, and thus the corresponding curves are superposed, this explain that it is hard to distinguish them on the figure. The performance of the algorithms in the overcomplete dictionary is always better than in the orthogonal dictionary, but the gain is less with MP than with the other algorithms (OMP,GP). Moreover, it is interesting to remark than the gain with OMP and GP is more important on the orchestra signal. The computation time needed by the three greedy algorithms are now given. Fig. B.2 shows computation time per iteration for the three algorithms. It is interesting to remark that the compuation time needed by one iteration of OMP becomes very high when the number of selected atoms increases, contrary to GP and MP which have a nearly constant computation time per iteration. It also interesting to remark than GP has almost the same computation time as MP.

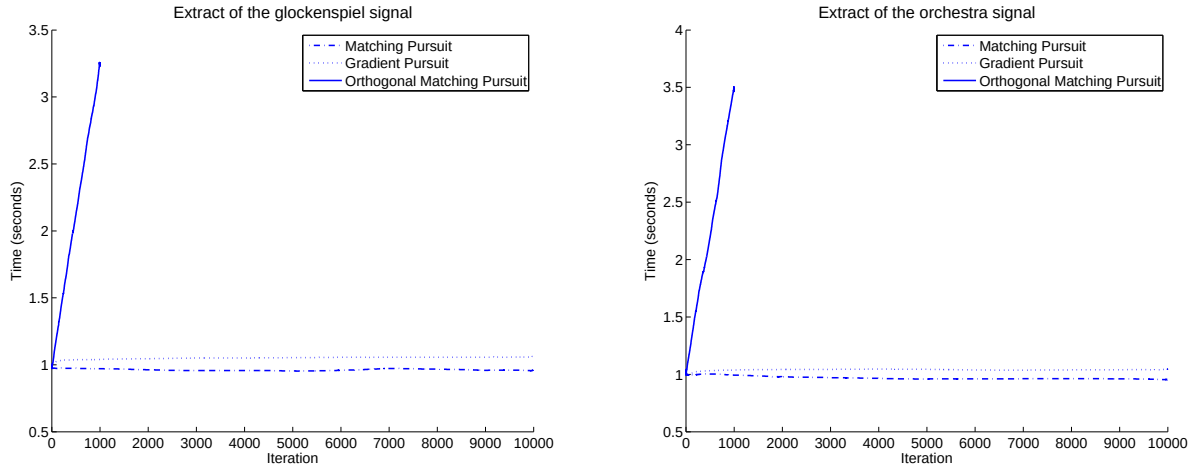

Figure B.2: *Computation time comparison of several greedy algorithms.*