Université de Nice-Sophia Antipolis

École doctorale STIC

THÈSE

pour obtenir le titre de

Docteur en Sciences

de l'UNIVERSITÉ de Nice-Sophia Antipolis

Mention : Automatique, Traitement du Signal et des Images

présentée par

Marie OGER

Model-based techniques for flexible speech and audio coding

Thèse dirigée par

Marc ANTONINI, Directeur de Recherche CNRS
Equipe d'accueil : CReATIVe, Laboratoire I3S, UNSA-CNRS

et

Stéphane RAGOT
Equipe d'accueil : TPS, Laboratoire SSTP, France Télécom R&D Lannion

soutenance prévue le 13 décembre 2007

Jury:

Robert M. Gray	Université de Stanford	Rapporteur
Pierre Duhamel	CNRS Paris	Rapporteur
Marc Antonini	CNRS Sophia Antipolis	Directeur de thèse
Stéphane Ragot	France Télécom Lannion	Directeur de thèse
Michel Barlaud	Université de Nice-Sophia Antipolis	Examinateur
Geneviève Baudoin	ESIEE Paris	Examinateur
Christine Guillemot	INRIA Rennes	Examinateur
Bastiaan Kleijn	KTH Stockholm	Examinateur

Acronyms

3GPP	3rd Generation Partnership Project
AAC	Advanced Audio Coding
AbS	Analysis-by -Synthesis
ACELP	Algebraic Code Excited Linear Prediction
ACR	Absolute Category Rating
ADPCM	Adaptive Differential Pulse Code Modulation
AMR-WB	Adaptive Multi Rate-Wideband
AMR-WB+	Extended Adaptive Multi-Rate Wideband
ATC	Adaptive Transform Codec
BER	Bit Error Rates
BSAC	Bit-Sliced Arithmetic Coding
CCR	Comparative Category Rating
CELP	Code Excited Linear Predictive
CMOS	Comparison Mean Opinion Score
DCR	Degradation Category Rating
DCT	Discrete Cosine Transform
DMOS	Degradation Mean Opinion Score
DPCM	Differential Pulse Code Modulation
DWT	Discrete Wavelet Transform
e-AAC+	Enhanced High Efficiency Advanced Audio Coding
EM	Expectation Maximization

FEC	Frame Error Concealment
FER	Frame Error Rate
FFT	Fast Fourier Transform
HE-AAC	High Efficiency AAC
GMM	Gaussian Mixture Model
i.i.d.	Independent and Identically Distributed
IP	Internet Protocol
ISDN	Integrated Services Digital Network
ISO	International Standardization Organization
ITU	International Telecommunications Union
JBIG	Joint Bi-level Image experts Group
JPEG	Joint Photographic Experts Group
KLT	Karhunen-Loeve transform
LPC	Linear Prediction Coding
LSB	Least Significant Bit
LSF	Linear Spectrum Frequency
LSP	Linear Spectrum Pair
LTP	Long-Term Prediction
MBMS	Multimedia Broadcast and Multicast Service
MDCT	Modified Discrete Transform
MLT	Modulated Lapped Transform
MMS	Multimedia Messaging Service
MOS	Mean Opinion Score
MPEG	Moving Picture Expert Group
M/S	Mid/Side
MSB	Most Significant Bit

MSE	Mean Square Error
MUSHRA	Multi Stimulus test with Hidden Reference and Anchors
NMR	Noise to Mask Ratio
OLQA	Objective Listening Quality Assessment
ONRA	Objective Noise Reduction Assessment
PCA	Principal Component Analysis
PDF	Probability Density Function
PEAQ	Perceptual Evaluation of Audio Quality
PESQ	Perceptual Evaluation of Speech Quality
PSS	Packet-Switched Streaming
QMF	Quadrature Mirror Filter
RAM	Random-Access Memory
ROM	Read-Only Memory
SBR	Spectral Band Replication
SPIHT	Set Partitioning In Hierarchical Trees
SPL	Sound Pressure Level
TCX	Transform Coded Excitation
TDAC	Time-Domain Aliasing Cancellation
TDBWE	Time-Domain BandWidth Extension
TNS	Temporal Noise Shaping
TwinVQ	Transform-domain Weighted Interleave Vector Quantization
VoIP	Voice over IP
VQ	Vector Quantization

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Avec l'évolution continue des technologies et des services de communication, les réseaux (d'accès et coeur) et les terminaux sont devenus très hétérogènes. En effet, les communications peuvent passer par le réseau téléphonique commuté (RTC), les liens mobiles (GSM, wi-fi, bluetooth), les modems (ADSL, FTTH,...), etc. De plus, on trouve des combinés téléphoniques en bande étroite, des téléphones HD en bande élargie, des soft phones (PC communiquant), des mobiles multi-usages, etc. En plus de cette hétérogénéité, on assiste actuellement à une importante prolifération des standards de codage de parole et audio. Ces standards ont été développés en majorité pour répondre aux besoins successifs du marché pour des applications de téléphonie fixe (G.711, G.729), téléphonie mobile (GSM-FR, HR, EFR, AMR-NR, WB, QCELP, EVRC), multimédia de masse (MPEG), visiophonie, visioconférence (G.722, G.722.1, G.722.1C), voix sur IP (G.723.1, G.729B, G.729.1).

Cette hétérogénéité des réseaux et des terminaux ainsi que la prolifération de standards spécifiques amènent naturellement à chercher de nouveaux modèles de codage qui soient généraux, adaptés à de multiples scénarios d'applications. En terme de technique de codage, la plupart des standards de codage de parole existants sont peu flexibles : ils utilisent notamment des tables de quantification vectorielle stockées, une allocation des bits fixe à certains paramètres de codage, et leurs performances sont souvent optimisées pour un certain point de fonctionnement (type de signal parole/musique, largeur de bande, débit). Il existe des algorithmes multi-débits - par exemple AMR-NB et -WB, VMR-WB ou G.729.1 - qui sont plus flexibles que les algorithmes à débit fixe mais ils ne font que combiner plusieurs codeurs. En codage audio, il existe de nombreux standards couvrant des configurations multiples (en débit, largeur de bande), tels que les codeurs MPEG-1/2 Layer III ou MPEG-4 AAC, mais ils ne sont adaptés qu'à des applications de stockage et diffusion, ils ne donnent pas une bonne qualité pour la parole à bas débit et nécessitent également un stockage des tables de quantification (codage de Huffman) ; le codage MPEG-4 BSAC a les mêmes inconvénients même s'il est plus adapté à l'hétérogénéité des réseaux de par sa nature hiérarchique.

L'objectif de la thèse est donc de développer des techniques de codage opti-

males - compétitives avec les méthodes de l'état de l'art - plus flexibles et pouvant s'adapter en temps-réel à différentes contraintes (débit, largeur de bande, retard, etc.). Pour se faire on se base sur différents outils fondamentaux : modélisation statistique (par GMM, modèles gaussiens généralisés), théorie débit-distortion de Bennett, codage entropique flexible comme le codage arithmétique, etc. Cette problématique générale de flexibilité de codage est reliée à une approche de codage basé modèle des coefficients LPC proposée récemment dans [104, 111]. Cette approche consiste à modéliser les coefficients LPC par un mélange de gaussiennes (GMM) afin d'optimiser la quantification. De même, on trouve en codage d'images et vidéo des travaux récents sur l'allocation des bits basée modèle [88]. Dans cette thèse, on vise à développer des techniques "basées modèle" similaires.

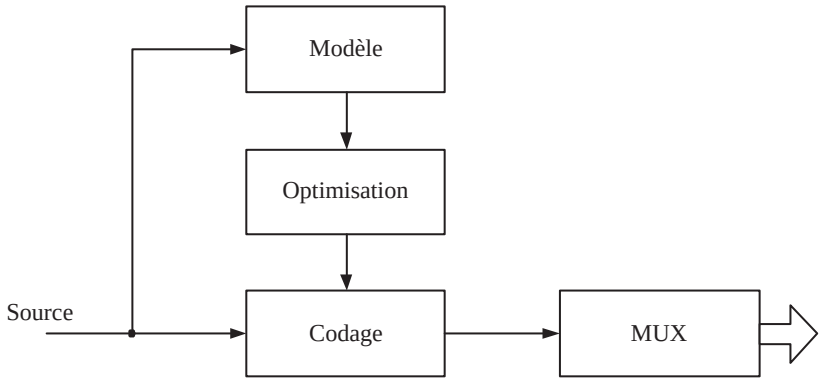


Figure 1.1: Principe du codage basé modèle.

La Fig. 1.1 présente le principe du codage basé modèle. Une source est approchée par un modèle (GMM, Gaussien généralisé, HMM, ...). On effectue une optimisation du codage basé sur la connaissance du modèle, puis on code la source. La source est ainsi codée suivant la fonction de distribution de probabilité (pdf) du modèle et les paramètres de codage sont optimisés en fonction du modèle. Cette modélisation permet d'avoir un codage "scalable" en débit, une complexité indépendante du débit et d'être adaptable à différentes sources (audio, parole, image, ...).

De plus cette thèse a été motivée par le lancement du projet européen Flexcode (Flexible Coding for Heterogeneous Networks) en juillet 2006, ayant pour but de développer de nouveaux algorithmes de codage conjoint source-canal qui sont plus flexibles que ceux existants tout en ayant une qualité égale à l'état de l'art. Dans le cadre de ce projet France Télécom est en particulier impliqué dans le WP1: codage de source, et sera chargé de développer des techniques de codage par transformée.

Dans le chapitre 2, nous présentons les standards de codage audio et parole

en bande élargie qui existent actuellement. Ces standards sont voués à des applications bien précises et on se propose de les classer en deux catégories: les codeurs conversationnels et ceux pour le stockage et la transmission d'information. La principale différence entre ces deux catégories étant le délai du codeur. Ces codeurs serviront de références aux codeurs que nous proposons.

Dans le chapitre 3, nous présentons des outils pour la mise en oeuvre d'un codeur flexible. En particulier, nous présentons les deux modèles utilisés dans nos travaux : le modèle Gaussien généralisé et le mélange de Gaussiennes (GMM). Nous rappellerons aussi l'allocation des bits dans le cas d'une source Gaussienne généralisée proposée dans [88]. De plus nous présentons le principe du codage arithmétique [62, 126] qui sera utilisé dans nos codeurs. Finalement, la procédure pour les tests d'écoute mis en oeuvre pour vérifier la qualité d'un codeur est présentée.

Dans le chapitre 4, nous présentons une technique de codage flexible basé modèle des coefficients LPC. Des méthodes de codage modélisant les coefficients LPC par un mélange de Gaussiennes (GMM) afin d'optimiser la quantification ont été proposées dans [104, 111]. Nous proposons d'améliorer ces techniques en approchant l'erreur de prediction des coefficients LPC par un modèle Gaussien généralisé. Nous comparons les performances du quantificateur proposé avec celui utilisé dans le codeur ITU-T G.722.2 [40] pour la quantification des coefficients LPC. Le quantificateur proposé présente des performances équivalentes tout en étant moins complexes, en pouvant s'adapter à différents débits et en ayant un coût de stockage faible.

Dans le chapitre 5, nous présentons un codeur par transformée utilisant le codage stack-run. Cette technique de codage est couramment utilisée en images [122, 121, 125] mais n'a jamais été utilisée pour le codage audio. Nous proposons donc un codeur dont les coefficients après la transformation MDCT sont modélisés par une Gaussienne généralisée. Le modèle Gaussien généralisé est utilisé pour l'allocation des bits dans le codeur et pour optimiser la taille de la zone morte dans le quantificateur scalaire. Les performances de ce codeur sont comparées avec celles du codeur ITU-T G.722.1 [39]. La qualité du codeur est meilleure que celle du codeur ITU-T G.722.1 à bas débit et équivalente à haut débit. Par contre, bien que l'utilisation du modèle permette de diminuer la complexité de l'allocation de débit, la complexité de calcul du codeur avec codage stack-run est plus importante que celle du codeur ITU-T G.722.1, tandis que le coût mémoire est faible. L'utilisation du modèle nous permet d'avoir un schéma de codage plus flexible et ayant une bonne qualité pour la parole et l'audio.

Dans le chapitre 6, nous présentons un codeur par transformée utilisant le codage par plan de bits [109, 90, 59]. L'avantage de cette technique de codage est d'être scalable en débit. Nous proposons d'utiliser le modèle Gaussien généralisé

afin d'initialiser les tables de probabilités du codage arithmétique utilisé dans le codage par plan de bits. Ainsi, les tables de probabilités utilisées dans le codage arithmétique sont plus proche de la source et les performances du codage en sont améliorées. Les performances de ce codeur sont comparées avec celle du codeur avec codage stack-run et celle du codeur ITU-T G.722.1. La qualité du codeur est inférieure à celle du codeur avec codage stack-run à bas débit et équivalente à haut débit. Par contre, la complexité de calcul est plus faible que dans le cas du codeur avec codage stack-run car la boucle de contrôle du débit n'est pas nécessaire.

Enfin, dans le chapitre 7, nous présentons un bilan des travaux effectués durant cette thèse et des perspectives pour la suite de ces travaux.

Chapter 2

State of art in speech and audio coding

Nowadays, many speech and audio coding standards are available. They are often designed for specific applications and constraints. Those coders were build in order to response to the successive needs of the market for conventional telephones (G.711, G.729), cellular networks (GSM-FR, HR, EFR, AMR-NR, WB, QCELP, EVRC), multimedia, videophone, videoconference (G.722, G.722.1, G.722.1C), or voice over IP (G.723.1, G.729B, G.729.1). Here, only wideband (50-7000 Hz) or super-wideband (50-14000 Hz) speech and audio standards are presented. Wideband and super-wideband coders are the most recent and ITU-T still have works in progress on those coders. Also, we compared our works with ITU-T G.722.1 [39].

This chapter is organized as follows. Main attributes of speech and audio coder are given in Section 2.1. In Section 2.2, coding methods are presented. Three classes of models have been chosen: analysis by synthesis coding which is the common model in speech coding, perceptual transform coding which is mainly for audio coding and *universal* coding for both speech and music coding. Then wideband and super-wideband standards for cellular networks are presented in Section 2.3. A review of streaming standards for audio coding is done in Section 2.4 before concluding in Section 2.5.

2.1 Attributes of Speech and Audio Coder

The main attributes of speech and audio coder are: bitrate and quality, complexity, frame size and algorithmic delay and channel-error sensitivity. In general there is a trade-off between all these attributes.

2.1.1 Bitrate, bandwidth and quality

The bitrate (in kbit/s) measures the efficiency of the coding algorithm. The bitrate of a coder can be fixed or variable. Lower bitrates, 800 to 4800 bit/s are used for secure telephony applications. For the last wideband or super-wideband codec the bitrate goes up to 64 kbit/s.

The quality is linked to the bandwidth of the input signal. We can classify speech and audio coders depending on the bandwith of the input signal (mono):

- In the telephone band, 300-3400 Hz, speech is muffled and there is not enough band for music.
- The wideband, 50-7000 Hz, has a quality for speech and music equivalent to AM radio.
- The super-wideband, 50-14000 Hz, has a good quality for both speech and music and is close to the FM band.
- The hifi band, 20-15000 Hz, has a good quality for both speech and music and is equivalent to the FM band.
- The 20-20000 Hz band is used for CD quality applications.
- The 20-24000 Hz band is used for "perfect" quality applications such as recording studio, cinema or DVD.

The first two bandwidth are typical for speech coding whereas the three others are common in audio coding. Fig. 2.1 presents the different bandwidth used for speech and audio coding.

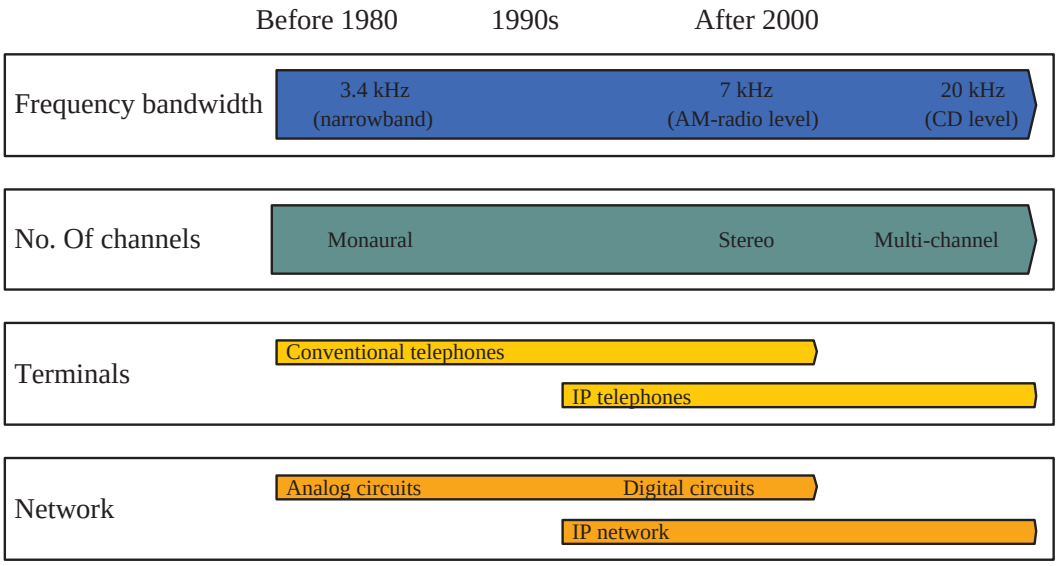


Figure 2.1: Bandwidth for different applications [48].

In order to measure the quality, subjective and objective tests are conducted. For speech subjective tests, ITU-T P.800 [81] describes different methods such as: absolute category rating (ACR), degradation category rating (DCR), comparative category rating (CCR). They yield to the mean opinion score (MOS), the comparison mean opinion score (CMOS) or the degradation mean opinion score (DMOS). For

audio signals ITU-T BS.1116 and BS.1534 [18, 20] propose methodologies for subjective test. The multi stimulus test with hidden reference and anchors (MUSHRA) or BS.1534 is a common quality test when there is important degradations in the signal [20].

In order to evaluate the objective quality of narrowband speech, the perceptual evaluation of speech quality [82, 83] can be used. The WB-PESQ score [84] is used to evaluate objectively the quality of a wideband speech coder. ITU-T P.OLQA and P.ONRA [93, 94] are new objective tests in progress in order to replace WB-PESQ score. For audio signals the objective quality of speech can be evaluated by "perceptual evaluation of audio quality" (PEAQ) [19]. Fig. 2.2 presents the quality of different coder model depending on the bitrate. As we can see, the quality of parametric coder saturate very quickly where the quality of CELP and waveform coder keeps increasing with the bitrate.

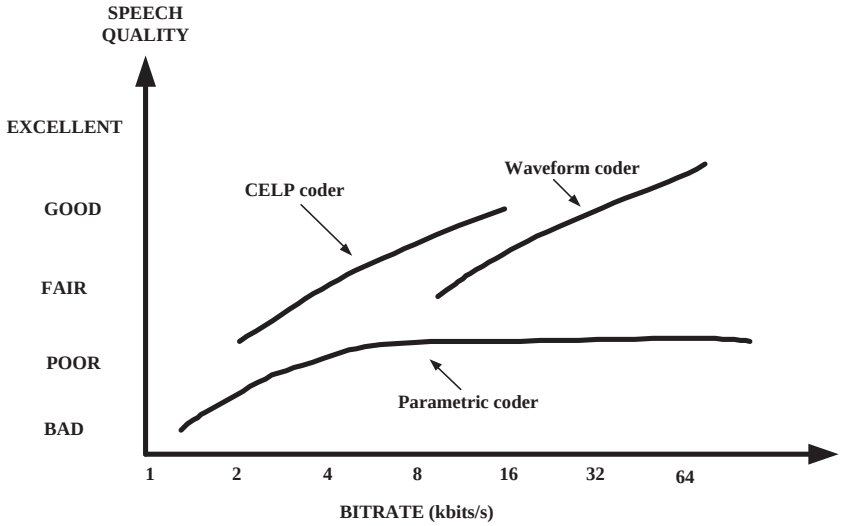


Figure 2.2: Quality of coder model.

2.1.2 Complexity

Complexity is a major parameter in practice. The computational complexity and memory requirement of a coder are related to the cost and power consumption of the hardware on which it is implemented. On a DSP chip, the algorithmic complexity is measured in terms of number of million instructions per seconds (MIPS), number of million cycles per second (MCPS), or weighted number of million operation per second (wMOPS) [36]. The RAM is necessary to store the intern variables of the coder (filter memory for example) and the ROM is necessary to store the constant and the instruction set of the program. They are expressed in number of bytes or words, note that $1\,kword = 1024\,word$ and $1\,Kword = 1000\,word$

2.1.3 Delay

In two-way communications, according to the E-model [35] if the end-to-end delay is between 0-150 ms it is acceptable for most of applications. If the end-to-end delay is between 150-400 ms it is acceptable for applications where interactivity is not important. If the end-to-end delay is more than 400 ms it is not acceptable for conversational tasks. Contrary to conversational applications, one-way communication (storage, streaming,...) do not have strong constraints on delay in general. Usually, the delay introduced by the coder is in the order of the framesize.

2.1.4 Channel-error sensitivity

In many applications the bitstream sent by the encoder may be corrupted by channel errors. There are two types of errors: bit errors or frame erasures (packet losses), both may occur in random or burst sequence. The robustness against bit errors can be improved by proper index assignment, the use of error detection/correction codes. Frame erasures are concealed using embedded coding, multiple description coding, or encoder-side redundancy. In practice conversational coders are often tested with 0.1 to 1 bits error rates (BER) and1% to 6% frame error rates (FER).

2.2 Speech and Audio Coding Model

We may classify speech and audio coding methods depending on the underlying coding model. We have chosen three classes of models that are current state of the art in speech and audio coding:

- Analysis by Synthesis (AbS) coding which is a linear predictive coding in which parameters are estimated by waveform matching [106] and is currently the most widely used speech coding approach.

- Perceptual transform coding which consists in a coder based on psychoacoustic principle [55, 54] and is the basis of audio coding.

- Universal speech/audio coding which is robust against the type of inputs (speech, audio). In general a multimode approach is used.

In addition we review some important tools:

- Parametric bandwidth extension algorithm [29, 52, 51] which is a process of expanding the frequency range (bandwidth) of a signal using low bit rate side information.

- Parametric Stereo coding [16, 17].

2.2.1 CELP coding

The code-excited linear-predictive (CELP) model is based on a source-filter model and is one of the most popular speech coding technique [106]. For a signal in telephone bandwidth, CELP coding has a good quality for bitrate from 6 to 16 kbit/s. For a wideband signal, CELP coding has a good quality for bitrate from 10 to 24 kbit/s. CELP is based mainly on the knowledge of the source (model) and on the receiver (perceptual filter).

This technique is based on an all-pole synthesis filter that models the short-term spectral envelope:

$$H(z) = \frac{1}{A(z)} = \frac{1}{1 + \sum_{i=1}^{n} a_i z^{-i}} \tag{2.1}$$

where n and a_i are respectively the model's order and the Linear Prediction Coefficient (LPC) [68, 73]. $A(z)$ is the prediction filter obtained using linear-prediction.

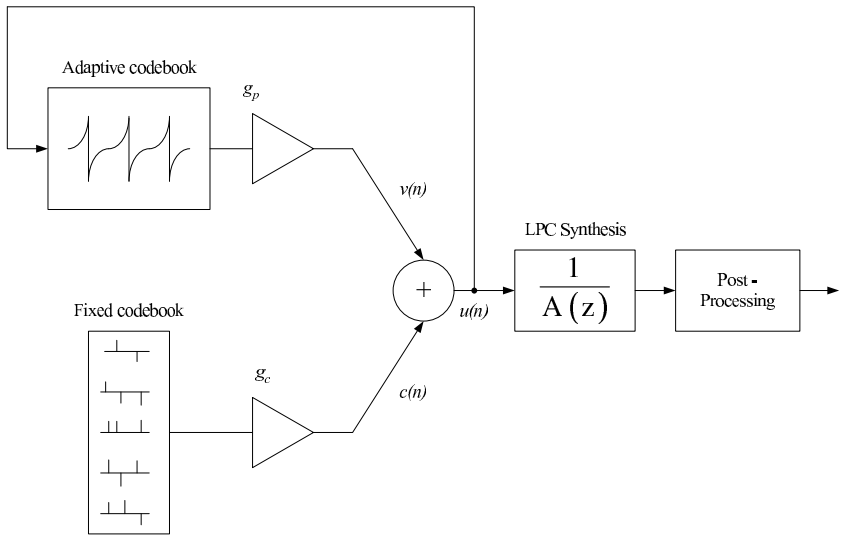


Figure 2.3: Block diagram of the CELP synthesis model.

The principle of the CELP synthesis model is presented in Fig. 2.3. The excitation, $u(n)$, is produced by summing the contributions from an adaptive (pitch) codebook, $v(n)$, and a fixed (innovative) codebook, $c(n)$. The best excitation, $u(n)$, is the one which minimizes the weighted mean square error between the original and the synthesized signal [86]. Post processing may be used to reduce coding noise and the idea is to shape the spectrum of the noise so that it follows the speech spectrum [10, 23].

The main principle behind CELP coding is analysis-by-synthesis (AbS) in order to encode the excitation. The encoding is done in three main steps:

- LPC coefficients are computed and quantized, usually as LSFs.

- The adaptive codebook gain is searched in order to have the best pitch combination using AbS. Algebraic coding methods are usually implemented to reduce the CELP codebook search complexity [9].
- The fixed codebook gain is searched for the best entry using AbS.

2.2.2 Perceptual transform coding

Perceptual transform coding

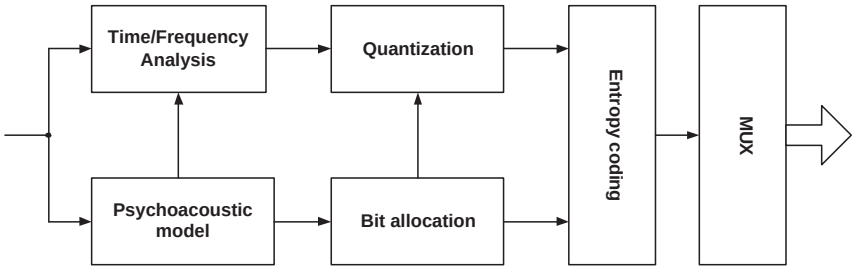


Figure 2.4: Block coding of perceptual transform coding.

Perceptual transform coding is used for high-quality audio coding [55, 54, 85] and the principles are presented in Fig. 2.4. In the reality, the human ear (receiver) is modeled mainly in frequency domain this is the main reason why frequency coding is used also frequency decomposition allows decorrelating signals components [53]. The quantization for the perceptual transform coder can be scalar or vectorial. A typical entropy coding is the Huffman coding after quantization. The time/frequency analysis is usually a short-term fast Fourier transform (FFT) or a modified discrete transform (MDCT). For MDCT, the coding is done on local sine bases. Psychoacoustic models [56, 55, 108], absolute hearing threshold, critical band frequency analysis or simultaneous masking, are used to estimate the signal masking power. The absolute threshold of hearing shown in Fig. 2.5 characterizes the amount of energy that can be detected by a listener in a noiseless environment. A typical measure of the absolute threshold is dB Sound Pressure Level (dB SPL). The noise to mask ratio (NMR) is a perceptual measurement which gives information about the distance between actual noise and masking threshold.

MDCT transform

The modified discrete transform (MDCT) [30], the modulated lapped transform (MLT) [71] or time domain aliasing cancellation (TDAC) [96] are equivalent transform. The MDCT is a lapped transform which is designed to perform on consecutive blocks [72]. The overlap between consecutive blocks is 50% as we can see in Fig. 2.6. In the absence of quantization, the reconstruction of the signal is perfect.

Two fast algorithms are typically used in order to calculate the MDCT:

- One is proposed by Duhamel and al. [30] based on complex FFT.

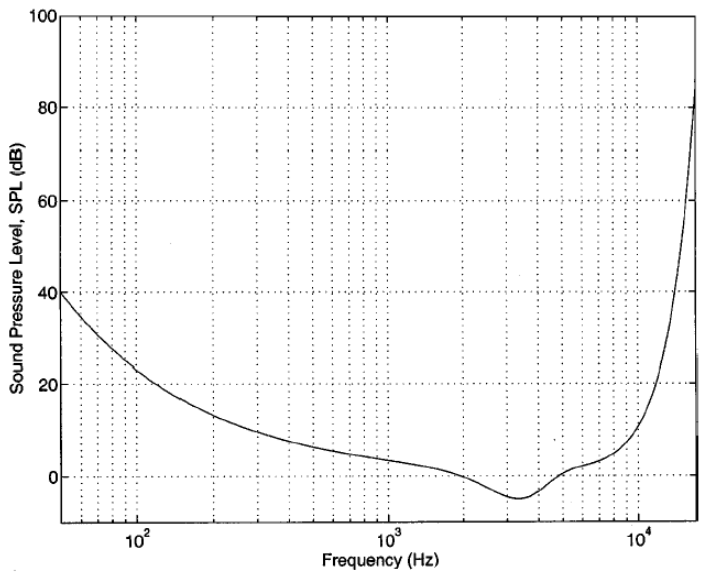


Figure 2.5: The absolute threshold of human hearing in quiet [55].

- Another is proposed by Malvar [71] based on DCT-IV.

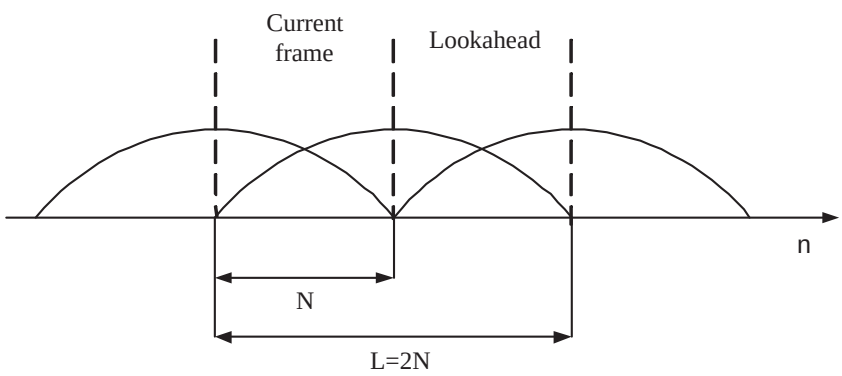


Figure 2.6: Principle of the MDCT.

We considered a signal $x(n)$ of $L = 2N$ dataset composed of N data of the current frame and N data of the future frame. After the MDCT transform presented in Fig. 2.6, we get a block $X(m)$ of N data. The TDAC and MLT algorithms are presented.

The TDAC [96] with sinusoidal window is defined as:

$$X(m) = \sum_{n=0}^{L-1} \frac{\sqrt{2}}{L} sin\left(\frac{\Pi}{L}(n+0.5)\right) cos\left(\frac{\Pi}{N}(n+N/2+0.5)(m+0.5)\right) x(n) \quad (2.2)$$

where $m = 0 \ldots N-1$.

The MLT [71] is defined as:

$$X(m) = \sum_{n=0}^{L-1} \frac{\sqrt{2}}{L} sin\left(\frac{\Pi}{L}(n+0.5)\right) cos\left(\frac{\Pi}{N}(n-N/2+0.5)(m+0.5)\right) x(n) \quad (2.3)$$

where $m = 0 \ldots N-1$.

2.2.3 Universal speech/audio coding

The concept of *universal* speech/music coder consists in reaching a given quality independent of the input signal (speech/music).

A combined adaptive transform codec (ATC) and coded-excited linear prediction (CELP), called ATCELP, has been proposed for the compression of wideband signals [25]. This codec proposed a switch between CELP and ATC coding model. CELP is mainly used for speech signals and ATC for music signals. The speech/music switch is done in such a way that the CELP is chosen for non stationary signals and ATC for stationary signals. The performance of this coder are close to ITU-T G.722 [38, 66].

Another coder which combined CELP coding and transform coding has been proposed in [116]. The speech/music discrimination is a combination of signal activity detection with a speech/non-speech discrimination. This coder improve the performance compared to single-mode coders.

The AMR-WB+ codec [7] is the last codec standardized for speech/music coding.

2.2.4 Parametric Bandwidth extension with side information

Because of the narrowband limitation for telephone network, the idea of applying an artificial bandwidth extension (BWE) in the receiver can be introduced [24, 52]. The BWE is an intermediate solution before the introduction to wideband telephone network.

For wideband speech coding, such as ITU-T G.722.2 [40], CELP coding is applied on the bandwidth up to 6400 Hz and artificial BWE for signal in 6400-7000 Hz band. This BWE is based on the transmission of linear prediction coefficients (LPC) with excitation [68]. For MPEG audio coding the BWE is based on spectral band replication (SBR) [29].

2.2.5 Parametric stereo coding

In order to do stereo coding there is a variety of method such as dual mono, mid/sid (M/S) coding [57], or intensity stereo coding [47]. Intensity stereo coding [47] is a joint stereo coding based on the principle of sound localization. It is supported by many audio compression formats such as MPEG-1 Layer III, or MPEG-2 AAC.

2.3 Wideband and Super-Wideband Coding Standards for Cellular Networks

In this section the wideband and super-wideband coding standards for cellular networks are presented. We restricted ourselves to wideband and super-wideband coding because it is the future of the network and also because we compared our works to wideband codec. Standards presented here were designed specifically for audio-videoconferencing, wideband telephony or voice over IP (VoIP) applications which means that the delay of the coder is an important parameter.

The wideband coders (G.722, G.722.1, G.722.2 and G.729.1) and the super-wideband (AMR-WB+, e-AAC+) coders are presented. A review of the ITU-T standards for narrowband has been done in [86, Chap.2].

ITU-T is working on G.711 coder [37] in order to do an extension of the narrowband into wideband. They are also working on a extension of G.729.1 [41] up to super-wideband, which is called for the moment EV-VBR.

2.3.1 ITU-T G.722

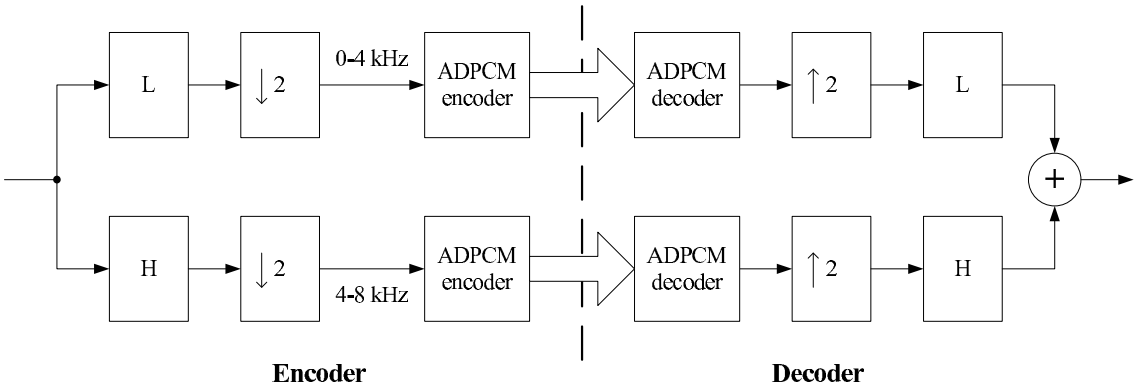


Figure 2.7: Block diagram of G.722.

G.722 was designed for different applications on ISDN, and is used in audio-videoconferencing, or wideband telephony. It is also a reference in term of coding quality for wideband speech coding.

G.722 [38, 66] has a subband ADPCM coding structure with different bitrate: 64, 56 or 58 kbit/s. Embedded ADPCM coding is used which give a scalable coder. The input signal is separated into two subbands by quadrature mirror filter (QMF) [33, 42, 123]: the low band [0, 4000Hz] and the high band [4000, 8000Hz]. The low band is in general more energetic and important perceptually than the high band, so it receives more bits. The high band always have a budget of 2 bits per sample where the low band have 4, 5, 6 bits per sample depending on the bitrate. Fig. 2.7 shows a simplified block diagram of the coder G.722.

Fig. 2.8 shows the frequency response of low- and high-pass filter of the QMF for G.722. The low- and high-pass decomposition filter (L and H) for the G.722

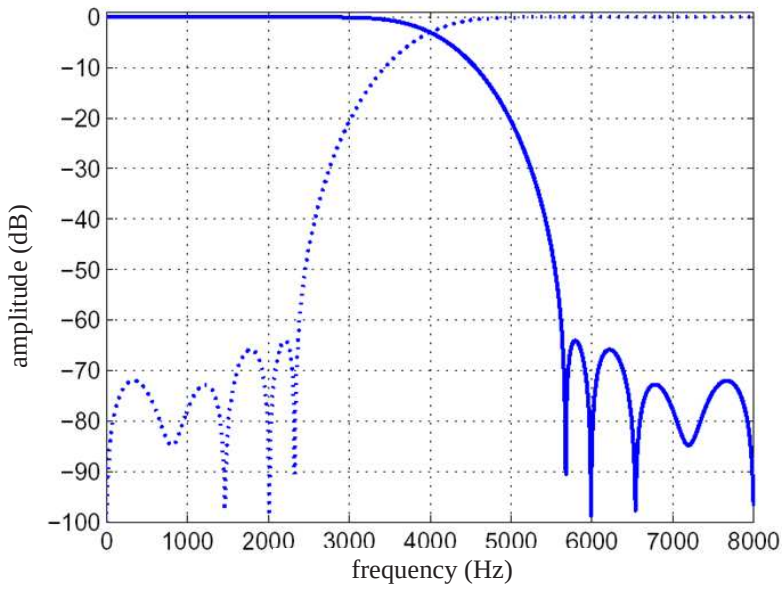


Figure 2.8: Frequency response of low- and high-pass QMF filters in G.722

have 24 coefficients. The overlap in the frequency domain is important. The delay introduced by the QMF is 23/16 (1.44) ms.

2.3.2 ITU-T G.722.1 (Siren 7) and G.722.1 Annex C (Siren 14)

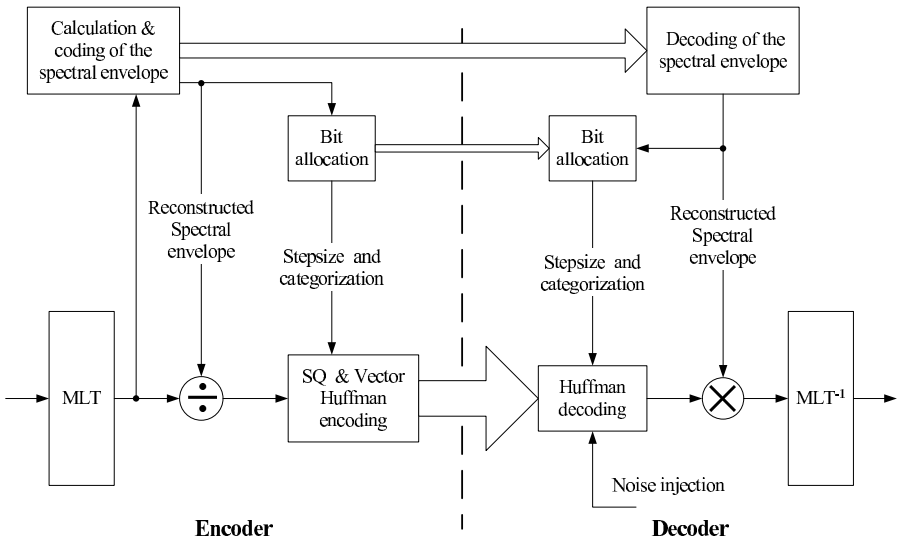


Figure 2.9: Block diagram of G.722.1.

G.722.1 is designed for videoconferencing [39]. It is recommanded for wideband

speech communication with few packet losses. The algorithm operates at 24 or 32 kbit/s. It is based on MLT transform coding with a frame of 20 ms (320 samples) and a lookahead of 20 ms, earlier versions used a discrete cosine transform (DCT) [27, 26].

Fig. 2.9 presents the coding scheme of G.722.1 coder. Each frame is coded independently. Every 20 ms, an MLT analysis is applied on 640 samples of the current and the future frame to obtain a spectrum of 320 coefficients.

The spectrum (50-7000 Hz) is divided into 14 subband of 20 coefficients each (500 Hz). Coefficients with frequency above 7 kHz are not coded. The bit budget for each frame is 480 or 640 depending on the bitrate (24 or 32 kbit/s). The spectral envelope is calculated and then coded by logarithmic scalar quantization with a stepsize of 3 dB and differential Huffman coding. This spectral envelope is equivalent to the energy of each subband in the MLT spectrum. Then a bit allocation is determined using a categorization procedure, 4 bits are sent to indicate the selected category. Each cotegory is associated to stepsize and Huffman tables [128]. Finally, the normalized MDCT coefficients are quantized by an uniform scalar quantizer with dead-zone and coded by Huffman coding.

G.722.1 Annex C [21] is an extension of G.722.1 for super-wideband signals (50-14000 Hz) with a sampling frequency of 32 kHz. The bit rate of this coder is 24, 32 or 48 kbit/s.

2.3.3 ITU-T G.722.2 (also 3GPP AMR-WB)

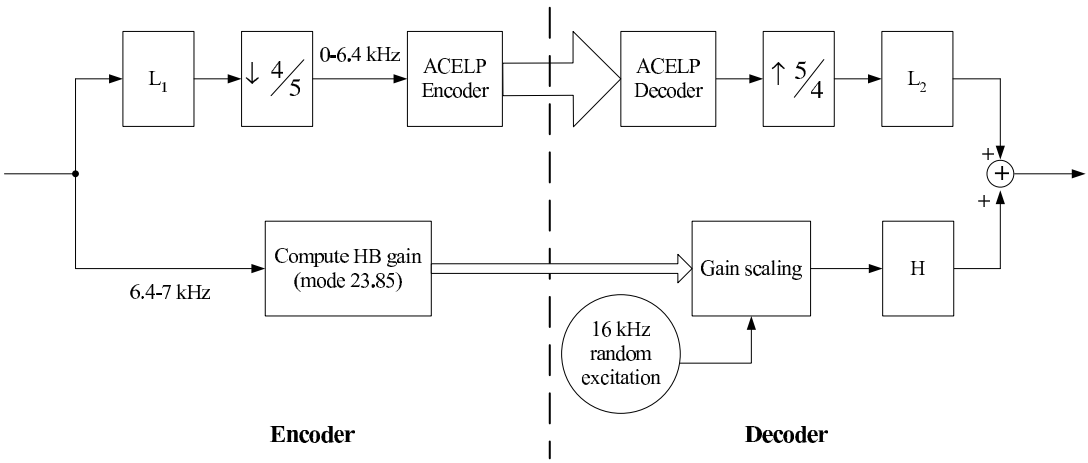


Figure 2.10: Block diagram of G.722.2.

G.722.2, or adaptive multi rate-wideband (AMR-WB) coder, has been designed for wideband telephony over wireless networks (GSM, UMTS) [40, 12].

G.722.2 is a multi-rate coder based on the algebraic code excited linear prediction (ACELP) technology [102]. AMR-WB coder consists of nine coding modes with bitrate of 23.85, 23.05, 19.85, 18.25, 15.85, 14.25, 12.65, 8.85 and 6.60 kbit/s. The

signal is sampled at 16 kHz and each frame has a size of 20 ms and a lookahead of 20 ms. ACELP coding is applied to the low band (50-6400 Hz) whereas in the high band (6400-7000 Hz) the signal is coded by with band extension as shown in Fig. 2.10. The bandwidth extension, except at 23.85 kbit/s, is done by a 16 kHz random excitation and no parameters are transmitted to the decoder. At 23.85 kbit/s high band gain are transmitted to the decoder using 4 bits per subframe.

Fig. 2.11 shows the frequency response of low- and high-pass filter (L_1/L_2 and H) of the G.722.2. The low-pass filters (L_1 and L_2) have 120 coefficients and the HF pass-band filter (H) has 31 coefficients.

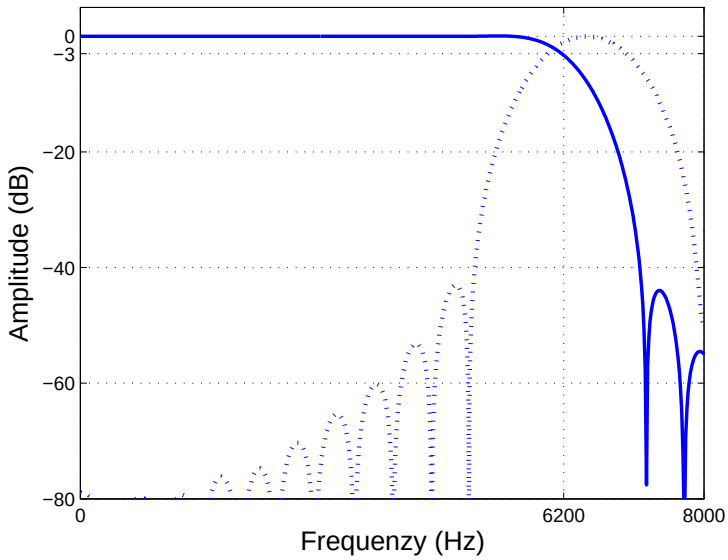


Figure 2.11: Frequency response of down/up- sampling and pass-band filter in G.722.2.

2.3.4 ITU-T G.729.1

G.729.1 is a scalable coder designed for wideband telephony and voice over IP (VoIP) applications [41, 97]. G.729.1 has 12 different bitrates from 32 to 8 kbit/s with wideband redundancy starting at 14 kbit/s and narrowband at 8 and 12 kbit/s.

Fig. 2.12 presents the encoding structure of G.729.1. The input signal is sampled at 16 kHz and each frame has a size of 20 ms. The input signal is split into two subbands using a QMF filter of 64 coefficients. The low-band signal is preprocessed by a high-pass filter of 50 Hz cutoff and then encoded by CELP coder. In order to improve the recovery of the decoder in case of frame erasures a frame error concealment (FEC) encoder is used. The high-band signal is preprocessed by a low-pass filter of 3 kHz cutoff and then is encoded by the time domain bandwidth extension (TDBWE) encoder [97].

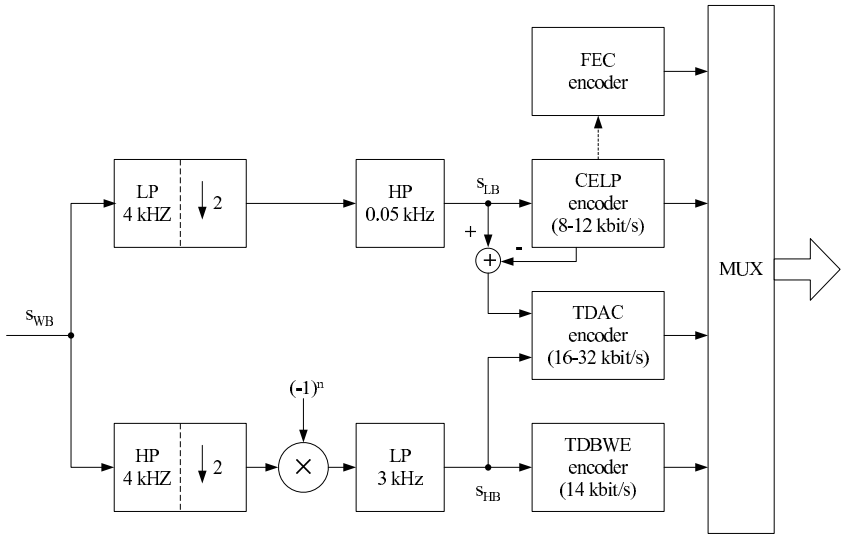


Figure 2.12: Block diagram of G.729.1 encoder.

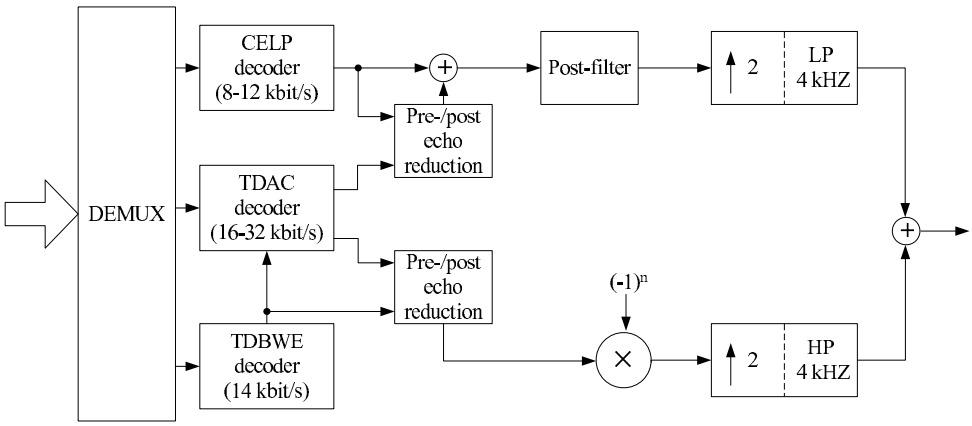


Figure 2.13: Block diagram of G.729.1 decoder (in the absence of frame erasures).

Fig. 2.13 presents the decoding structure of G.729.1. The decoder operates in an embedded manner depending on the received bit-rate [97].

2.3.5 AMR-WB+ codec

The AMR-WB+ codec is recommended for PSS, MMS, and MBMS services. In low bit-rate AMR-WB+ performs better with speech dominated content than e-AAC+ [124].

The AMR-WB+ codec [74, 7] is based on the ACELP/TCX [102, 63] model which switches between LP-based and transform-based coding depending on the signal characteristics. A variant of AMR-WB is used in the ACELP mode and TCX with algebraic VQ [98] is used in the transform coding mode.

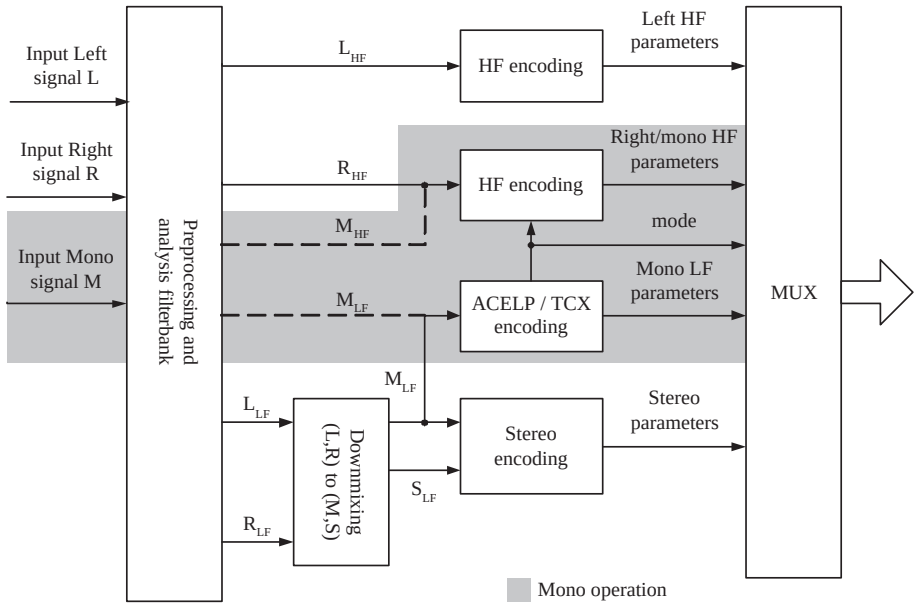


Figure 2.14: Block diagram of AMR-WB+ encoder.

Fig. 2.14 presents the block diagram of AMR-WB+ encoder. A bandwidth extension is done on the high band for all frequencies above the intermediate frequency 6.4 kHz. An hybrid ACELP/TCX model is applied on the low band.

2.3.6 e-AAC+ codec

Enhanced AAC+ (e-AAC+) performs better than AMR-WB+ with audio database at low bitrate and in high bitrate with speech database [124]. E-AAC+ is very close to MPEG-4 HE-AAC v2 [6].

E-AAC+ [8] is a combination of AAC [3], Spectral Band Replication (SBR) [29] and Parametric Stereo (PS) [34, 108, 107].

Fig.2.15 shows the general structure of e-AAC+ encoder. The PS is used for low bitrate coding and stereo bitrate at and below 44 kbit/s. Above 44 kbit/s, AAC with mid/sid (M/S) stereo is applied on the low band signal, and SBR is applied in high band.

2.4 Streaming Standards for Audio Coding

In this section the wideband streaming standards for audio coding are presented. We focused on streaming standards such as MPEG-1/2 Layer III and MPEG-4. MPEG-4 has three quantization and coding techniques and the most important ones are presented: AAC and BSAC. MPEG is working on AAC enhanced low delay (ELD) for applications with a critical delay.

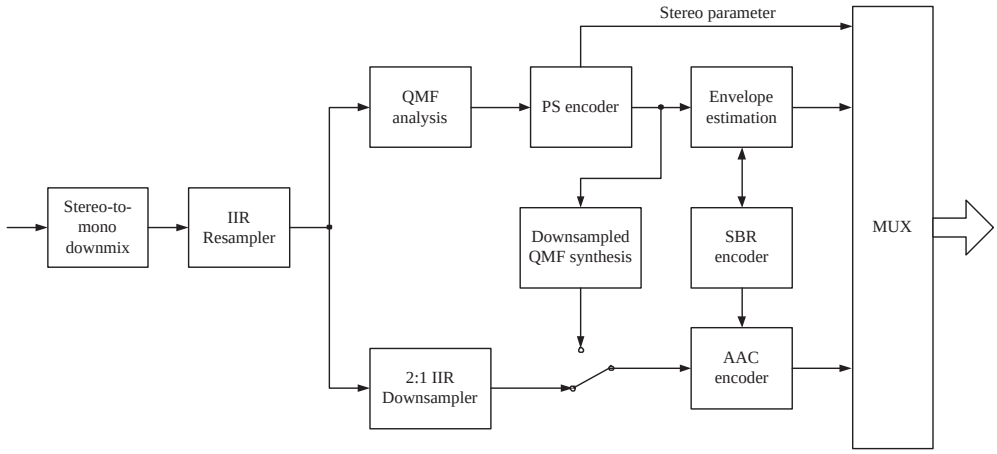
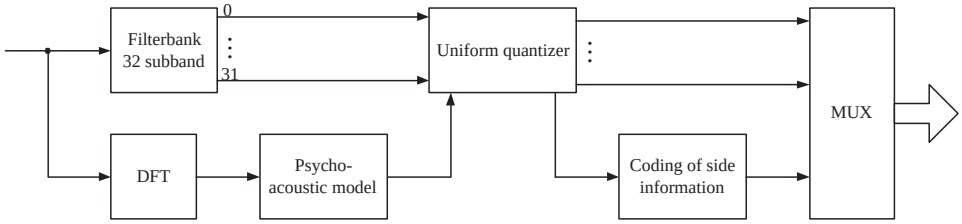


Figure 2.15: Block diagram of e-AAC+ encoder.

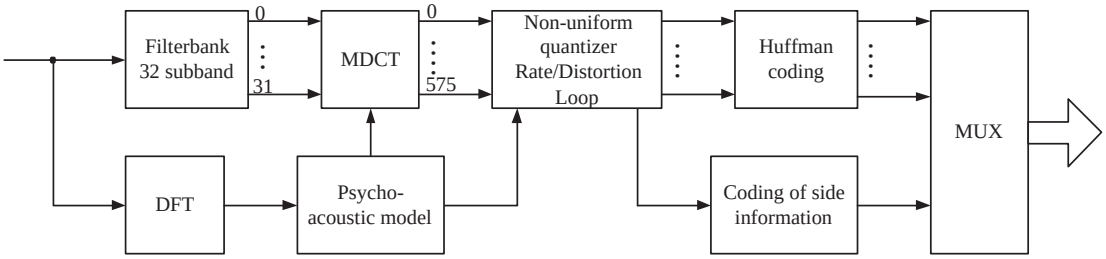
2.4.1 MPEG-1/2 Layer III

The MPEG-1 audio standard specifies three layers, each layer is a different compression method and the complexity and performance are increased each time. Also, MPEG-1 Layer III can decode audio files compressed by Layer I or II.

MPEG-1 Layer I and II encoder is presented in Fig. 2.16 (a). The filterbank is a polyphase filter of 32 subbands and then an uniform quantization is applied on the signal [14, Chap.11].



(a) Block diagram of MPEG-1 Layer-I and II encoder.



(b) Block diagram of MPEG-1 Layer-III encoder.

Figure 2.16: Principle of GMMM-based LSF vector quantization.

MPEG-1/2 Layer III [15, 1, 2] [14, Chap.11] also knows as MP3 has become the

most widely used coding format for music on Internet. The size of audio files are limited and so the downloading time on Internet. The recommended bit rate is 32 to 320 kbit/s for MPEG-1 Layer III and 8-160 kbits/s for MPEG-2 Layer III. The block diagram of MPEG-1 and 2 Layer III is presented in Fig. 2.16 (b).

The filterbank is an hybrid filterbank which is a cascade of two filterbanks. First, a polyphase filterbank already used in MPEG-1 Layer II and III is applied on the signal. Then a MDCT transform of 36 coefficients (12 samples for short window) is used in order to have a better compression level [14, Chap.11].

A rate-distortion optimization is done with a system of two nested iteration loops for the quantization [103, 15]. Quantization is done via a power-law quantizer. Thus, bit are set in order to reduce the quantization noise. The quantized values are coding by Huffman coding.

2.4.2 MPEG-4 AAC,BSAC and AAC+

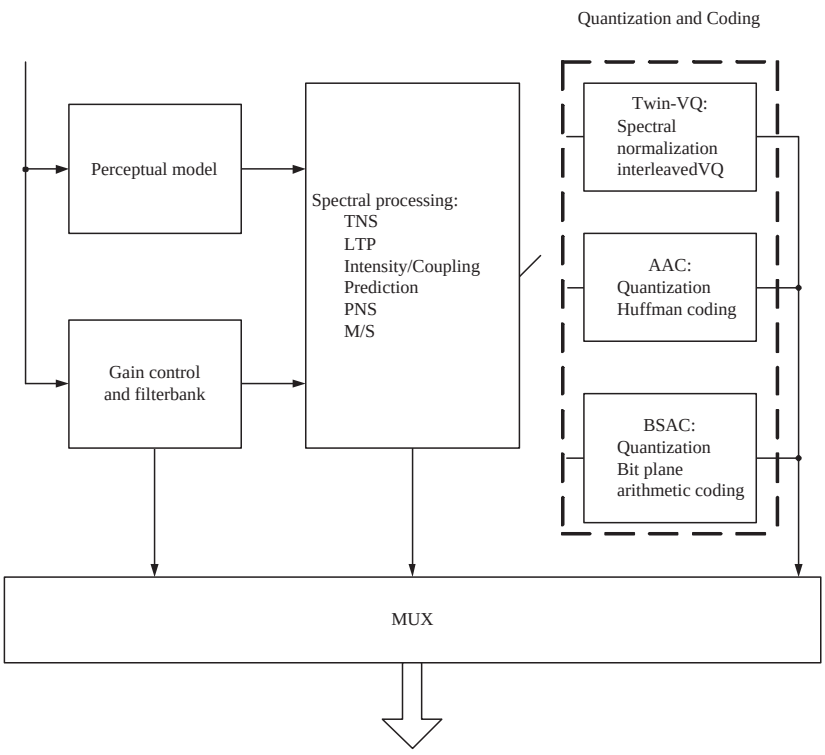


Figure 2.17: Block diagram of MPEG-4 encoder.

MPEG-4 has been designed for multimedia framework applications, mobile telephony or streaming [4]. The recommended bit rate is 2 to 64 kbit/s per channel [59, 4]. The block diagram of MPEG-4 is presented in Fig. 2.17.

The quantization of the parameters can be done in 3 ways (Twin VQ, AAC or BSAC). We focus here on the advanced audio coding (AAC) and the bit slice

arithmetic coding (BSAC).

MPEG-4 AAC and extension AAC+ (or HE-AAC)

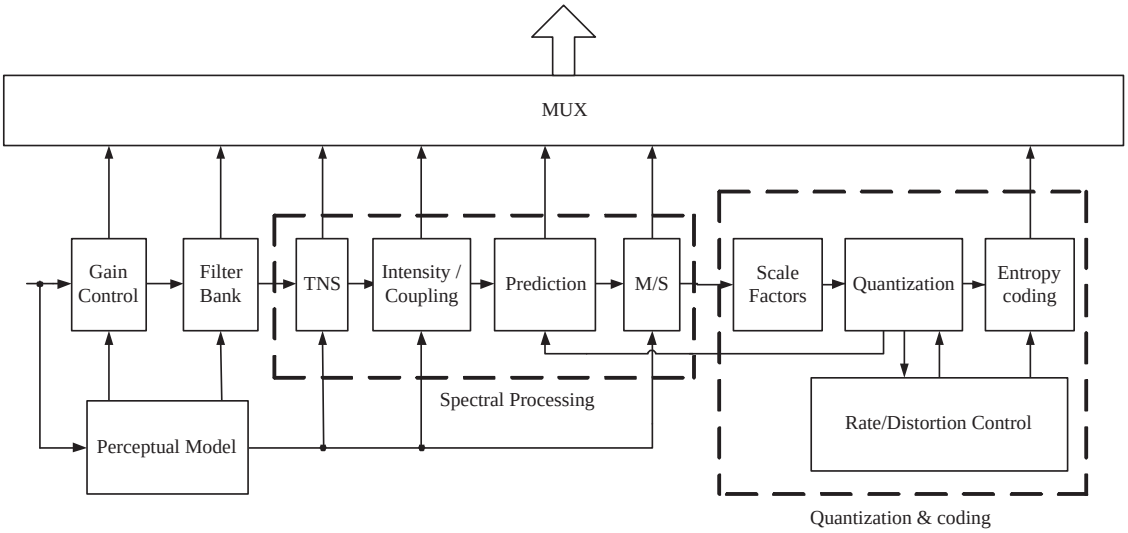


Figure 2.18: Block diagram of MPEG-2/4 AAC encoder.

AAC is a perceptual encoder that supports up to 48 channels with sampling rates from 8 to 160 kbit/s per channel [15, 3, 13][14, Chap.13,15]. The block diagram of MPEG-4 AAC is presented in Fig. 2.18.

Instead of having a cascade of two filters (polyphase + MDCT) as in MPEG-1/2 Layer III, AAC uses a MDCT with 50% overlap in the filterbank block. A non-uniform scalar quantization is applied on MDCT coefficients followed by Huffman coding similar to MP3. Tools such as temporal noise shaping (TNS), long-term prediction (LTP) or phase-noise suppression (PNS) are used in order to improve the signal compared to MPEG-1 Layer III [14, Chap.13,15].

TNS does noise shaping in time domain and improve the speech quality at low bit-rate. Instead of transmitting the left and right signals, the normalized sum (M as in Mid) and difference signals (S as in Side) are transmitted.

MPEG-4 AAC+, or High Efficiency AAC (HE-AAC), is an extension of AAC in order to reach high-quality at lower bitrates [5, 127]. AAC+ is using the AAC for the low frequency and spectral band replication (SBR) [29, 32] for high frequency as show in Fig. 2.19.

In the SBR encoder the control parameters are estimated so that the high frequency reconstruction in the reconstructed highband is perceptually as similar as the original highband.

MPEG-4 BSAC

BSAC is a coder similar to AAC expect that the Huffman coding is replaced by bit sliced arithmetic coding [90, 59]. The coding process is identical to AAC except

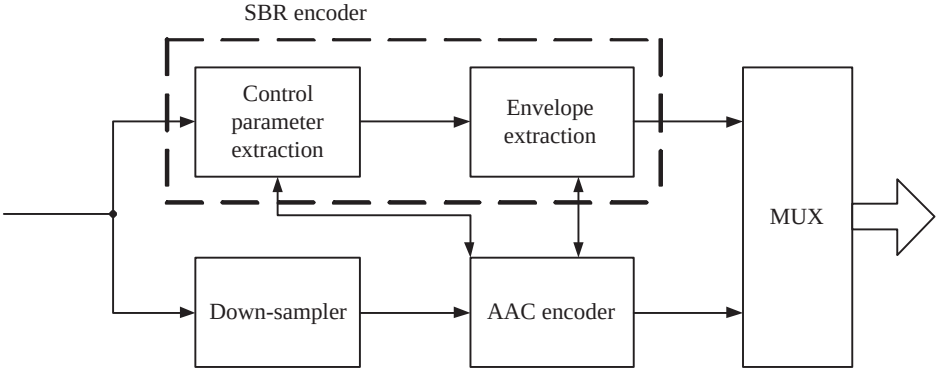


Figure 2.19: Block diagram of MPEG-4 HE-AAC encoder.

for the Huffman coding. The bitrate of the BSAC goes from 24 to 48 kbit/s per channel and it has a fine grain scalability (1 kbps/channel) which is not the case of TwinVQ and AAC.

To obtain such fine grain scalability a bit-sliced coding is applied on the quantized spectral data. An example of bit-sliced coding is presented in Fig. 2.20.

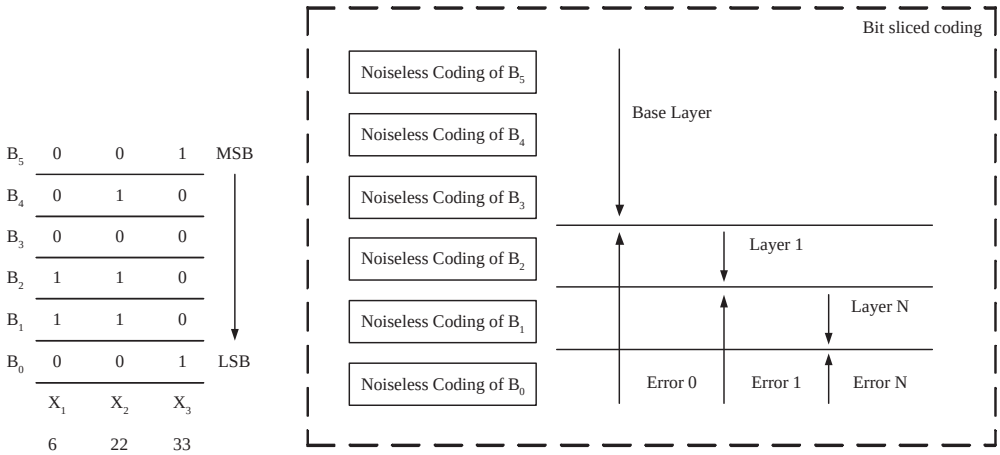


Figure 2.20: Example of Bit sliced Coding.

The quantized spectral data, X_i are represented in binary format, B_j. The organization of binary formatted data can be rearranged to bit-sliced format shown in Fig. 2.20. The MSB (Most Significant Bit) are more important than the LSB (Least Significant Bit) so the coding of MSB is a priority on LSB. Thus, the bit-sliced data is coded from the MSB to the LSB [90, 59].

2.5 Conclusion

In this chapter a review of wideband and super-wideband standard for streaming or cellular network applications has been done. Those standards are based on three models: analysis by synthesis coding which is widely used for speech coding, perceptual transform coding which is for audio coding and *universal* coding for both speech and music coding. Our objectives is to present a model-based coding method with the same performance than the standards but with more flexibility in terms of applications, bitrates, signals (speech/music).

Chapter 3

Tools for Flexible Signal Compression

In this chapter we are presenting different methods and models for flexible signal compression.

This chapter is organized as follows. In Section 3.1 the definition of scalar quantization and the distortion which measure the performance of a quantizer is given. Two special cases of scalar quantizer are also presented: the uniform scalar quantizer and the scalar quantizer with deadzone. Then in Section 3.2 the generalized Gaussian model is defined. This model is one of the statistical model used in order to approximate a source in our work. In the following Section 3.3 the Gaussian mixture model is presented. This model is used in order to classify linear-predictive coding (LPC) parameters. In Section 3.4, two model-based bit allocations are explained, one in case of high rate assumption and the other with the exact formula. Those two methods supposed that the pdf of the source can be modeled by a generalized Gaussian distribution. In Section 3.5 the principles of arithmetic coding are presented. Finally in Section 3.6 methodologies for subjective tests for speech and audio coder are given.

3.1 Scalar Quantization

Quantization is the basis of digital signal compression [45, 43]. For a scalar source x in $\mathbb{R}$, a *scalar quantizer* of $L \geq 1$ levels is defined by a coder C and a decoder D :

$$C : \left\{ \begin{array}{lcl} \mathbb{R} & \rightarrow & I = \{0, \ldots, L-1\} \\ x & \rightarrow & C(x) \end{array} \right. \quad D : \left\{ \begin{array}{lcl} I & \rightarrow & \mathbf{C} = \{y_0, \ldots, y_{L-1}\} \\ i & \rightarrow & D(i) \end{array} \right. \tag{3.1}$$

where $\mathbf{C} = \{y_0, \ldots, y_{L-1}\}$ is the finite set of values available at the decoder or codebook. The output values y_i are called the quantization level or codeword. Thus, $\hat{x} = C(x) = y_i$. The transmission rate, R, is given as $R = \log_2 L$ bits per sample.

3.1.1 Distortion

For a stationnary source x of pdf $p(x)$, the performance of the quantizer is usually measured by the average distortion:

$$D = E\left[d\left(x,\, C(x)\right)\right] = \int_{\mathbb{R}} d\left(x,\, C(x)\right)\, p(x)\, dx \tag{3.2}$$

where $E[.]$ is the expectation. The most commonly used distortion measure is the square error defined by:

$$d(x, \hat{x}) = (x - \hat{x})^2 \tag{3.3}$$

The mean square error (MSE) is defined as:

$$D = E\left[(x - C(x))^2\right] = \sum_{i \in I} \int_{R_i} (x - y_i)^2\, p(x)\, dx \tag{3.4}$$

where R_i is the cell decision defined as:

$$R_i = \{x \in \mathbb{R}^n | C(x) = i\} \tag{3.5}$$

where $R_i \bigcap R_j = \emptyset$ if $i \neq j$. The MSE distortion is simple to use but it does not match well subjective quality for speech or audio signals.

3.1.2 Uniform scalar quantization

A uniform scalar quantizer have cells decision R_i of regular size. If we note the stepsize q, we have $y_i - y_{i-1} = q$ and $y_i = (x_i + x_{i-1})/2$ for $i = 0, \ldots, L-1$. Fig 3.1 represents the staircase character of the uniform scalar quantizer.

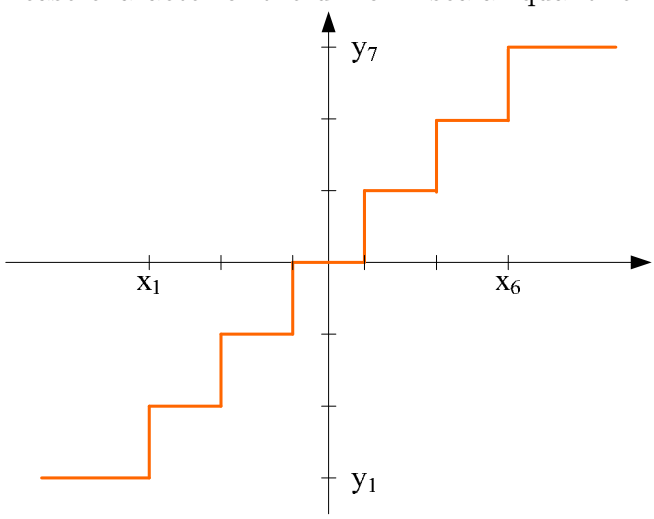
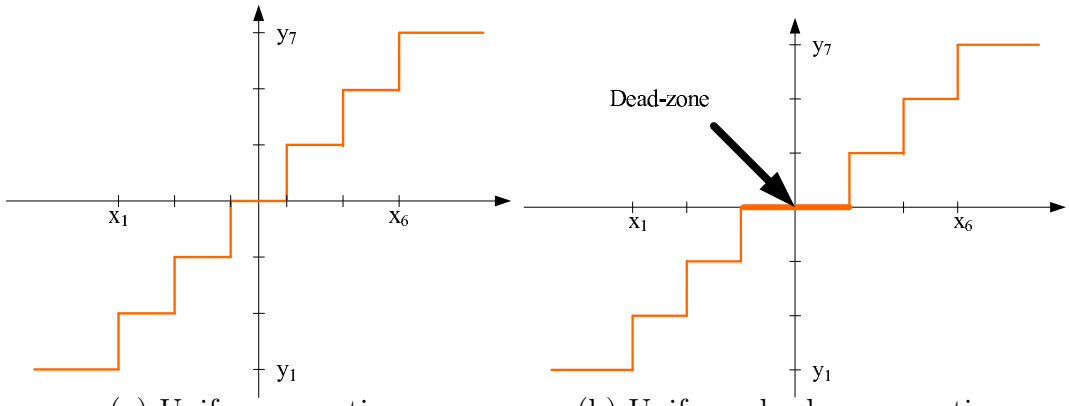


Figure 3.1: Staircase Character of the uniform quantizer

The average distortion of the uniform scalar quantizer is defined as $D = q^2/12$ in case of high rate assumption.

3.1.3 Uniform scalar quantizer with deadzone

The deadzone is the Voronoi region around zero. Fig. 3.2 presents the difference between a uniform scalar quantizer and a uniform scalar quantizer with deadzone.



(a) Uniform quantizer (b) Uniform deadzone quantizer

Figure 3.2: Difference between scalar quantizer with or without deadzone.

It is possible to estimate the optimal size on the deadzone with the pdf of the source. This estimation is presented in Section 3.4.

The uniform scalar deadzone quantization for a given signal x is given by:

$$\begin{cases} x \in (-z/2, z/2) & \hat{x} = 0 \\ x \in (z/2 + kq, z/2 + (k+1)q) & \hat{x} = y_k \\ x \in (-z/2 - (k+1)q, -z/2 - kq) & \hat{x} = y_{-k} \end{cases}$$

where $k \in \mathbf{N}$, $\hat{x}$ is the quantized signal and y_k are the quantization levels.

3.2 Generalized Gaussian modeling

Generalized Gaussian modeling is a statistical model applied to represent a given source. The information given by this model are used in order to improve the coding performance of the source. Generalized Gaussian modeling is commonly used in image and video coding [58, 88]. However, its application to speech and audio coding is quite new.

The definition of the generalized Gaussian pdf is given in Section 3.2.1. A review of estimation methods for the shape parameter, α, of the generalized Gaussian model is given in Section 3.2.2. Then estimation examples on speech are presented in Section 3.2.3.

3.2.1 Definition of the generalized Gaussian pdf

The probability density function (pdf) of a zero-mean generalized Gaussian random variable z of standard deviation σ is given by [89]:

$$g_{\sigma,\alpha}(z) = \frac{A(\alpha)}{\sigma} e^{-|B(\alpha)z/\sigma|^{\alpha}}, \tag{3.6}$$

where α is a shape parameter describing the exponential rate of decay and the tail of the density function. The parameters $A(\alpha)$ and $B(\alpha)$ are given by:

$$A(\alpha) = \frac{\alpha B(\alpha)}{2\Gamma(1/\alpha)} \qquad \text{and} \qquad B(\alpha) = \sqrt{\frac{\Gamma(3/\alpha)}{\Gamma(1/\alpha)}}, \tag{3.7}$$

where $\Gamma(.)$ is the Gamma function defined as:

$$\Gamma(\alpha) = \int_0^\infty e^{-t} t^{\alpha+1}\, dt. \tag{3.8}$$

The Laplacian and Gaussian distributions correspond to the special case $\alpha = 1$ and 2 respectively:

$$\text{Laplacian distribution}: \; g_{\sigma,\alpha=1}(x) = \frac{1}{\sqrt{2}\sigma} e^{\frac{\sqrt{2}|x|}{\sigma}} \tag{3.9}$$

$$\text{Gaussian distribution}: \; g_{\sigma,\alpha=2}(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{x^2}{2\sigma^2}} \tag{3.10}$$

The generalized Gaussian model is useful to approximate symetric unimodal distributions. Fig. 3.3 represents pdf of zero-mean generalized Gaussian distribution with different shape parameter. As we can see the smallest the shape parameter is, the most peaky the distribution is.

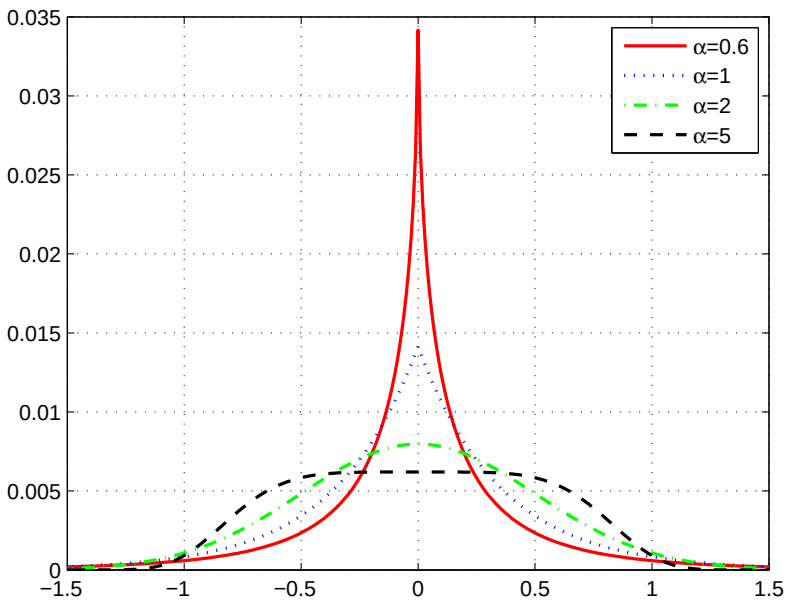


Figure 3.3: Example of probability density function

3.2.2 Estimation of the shape parameter α

Estimation methods of the shape parameter α are reviewed here. We classify estimation methods in "closed loop" and in "open loop". "Closed loop" methods estimate α by minimizing a distance criterion between data and model, while "open loop" methods provide an estimate of α without any distance criterion.

We assume that we have given samples $\{z_1, z_2, \ldots, z_N\}$ from a random variable z of pdf $g(z)$. We assume that $g(z)$ can be modeled by a generalized Gaussian. We estimate the shape parameter α of a generalized Gaussian model $g_{\sigma,\alpha}(z)$ approximating $g(z)$. To do so the normalized histogram $\tilde{p}(z)$ of $z_1, z_2, \ldots, z_N$ is compared with a sampled version of $g_{\sigma,\alpha}(z)$. The sampling step size is defined as a constant q. The estimation procedure is illustrated in Fig. 3.4. Without loss of generality the random variable z is supposed to be of unit variance and zero-mean.

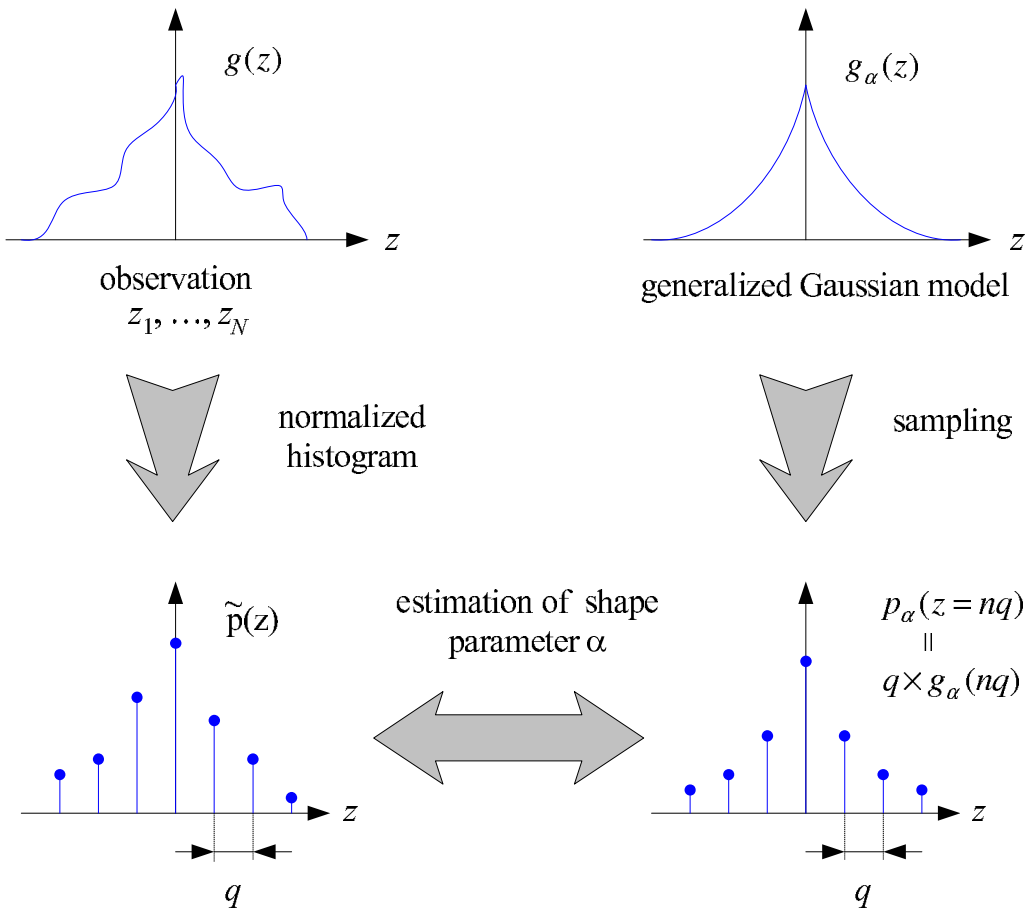


Figure 3.4: Estimation procedure for the shape parameter α

3.2.2.1 Estimation methods in closed loop

χ^2 statistics [87]

The χ^2 statistics evaluates a kind of distance between two probability density functions. We use here a χ^2 distance given by:

$$\chi^2(\alpha) = \sum_{z=\ldots,-q,0,q,\ldots} \frac{\left(\tilde{p}(z) - p_\alpha(z)\right)^2}{\tilde{p}(z) + p_\alpha(z)}, \tag{3.11}$$

where $p_\alpha(z) = g_\alpha(z) \times q$.
The estimated shape parameter is obtained by minimization of:

$$\hat{\alpha} = \arg\min_\alpha \chi^2(\alpha) \tag{3.12}$$

Kolmogorov-Smirnov statistic [87]
The Kolmogorov-Smirnov statistic is defined as:

$$KS(\alpha) = \max_z |G(z) - G_\alpha(z)| \tag{3.13}$$

where $G(z)$ and $G_\alpha(z)$ are the distributions:

$$G(z) = \sum_{n=-\infty}^{z/q} \tilde{p}(nq) \qquad \text{and} \qquad G_\alpha(z) = \sum_{n=-\infty}^{z/q} p_\alpha(nq) \tag{3.14}$$

The estimated shape parameter $\hat{\alpha}$ is found by minimization of:

$$\hat{\alpha} = \arg\min_\alpha KS(\alpha) \tag{3.15}$$

Kullback-Leibler divergence [61]
The Kullback-Leibler divergence is given by:

$$D(\tilde{p}||p_\alpha) = -\sum_{z=\ldots,-q,0,q,\ldots} p_\alpha(z) \log \frac{\tilde{p}(z)}{p_\alpha(z)} \tag{3.16}$$

The estimated shape parameter $\hat{\alpha}$ is obtained by minimizing this measure between the histogram $\tilde{p}$ and the generalized Gaussian model:

$$\hat{\alpha} = \arg\min_\alpha D(\tilde{p}||p_\alpha) \tag{3.17}$$

3.2.2.2 Estimation methods in open loop

Several methods are omitted here, e.g. ML estimation [58].

Estimation based on kurtosis (κ_α)
For a generalized gaussian source z of shape parameter α, the kurtosis κ_α is given by [87]:

$$\kappa_\alpha = \frac{E(z^4)}{E(z^2)^2} = \frac{\Gamma(5/\alpha)\Gamma(1/\alpha)}{\Gamma(3/\alpha)^2} \tag{3.18}$$

It can be verified that $\log \kappa_\alpha$ is approximatively a linear function of $1/\alpha$ [89]:

$$\log \kappa_\alpha \approx \frac{1.447}{\alpha} + 0.345 \tag{3.19}$$

Based on this approximation, the shape parameter α can be estimated as [89]:

$$\hat{\alpha} \approx \frac{1.447}{\ln \hat{\kappa} - 0.345}, \tag{3.20}$$

where $\hat{\kappa}$ is estimated from the data:

$$\hat{\kappa} = \frac{\frac{1}{n}\sum_{i=1}^{n} z_i^4}{\left(\frac{1}{n}\sum_{i=1}^{n} z_i^2\right)^2} \tag{3.21}$$

Method proposed by Mallat [69]

A relation between the variance $E(z^2)$, the mean of the absolute value $E(|z|)$ and the shape parameter α is given by [89]:

$$\frac{E(|z|)}{\sqrt{E(z^2)}} = \frac{\Gamma(2/\alpha)}{\sqrt{\Gamma(1/\alpha)\Gamma(3/\alpha)}} = F(\alpha) \tag{3.22}$$

The shape parameter α can be estimated as:

$$\hat{\alpha} = F^{-1}\left(\frac{\hat{m}_1}{\sqrt{\hat{m}_2}}\right) \tag{3.23}$$

where $\hat{m}_1 = \frac{1}{n}\sum_{i=1}^{n} z_i^2$ and $\hat{m}_2 = \frac{1}{n}\sum_{i=1}^{n} |z_i|$.

Estimation based on differential entropy (h_α)

The differential entropy of a generalized Gaussian distribution is given by [89]:

$$h_\alpha = \frac{1}{2}\log_2\left[\frac{4\Gamma(1/\alpha)^3}{\alpha^2\Gamma(3/\alpha)}\right] + \frac{1}{\alpha \log 2} \tag{3.24}$$

Based on high rate quantization theory, it can be shown that the entropy rate R of the quantized random variable z and the differential entropy $h(z)$ are related as follows [45]:

$$R \approx h(z) - \log_2 q \tag{3.25}$$

The estimated shape parameter $\hat{\alpha}$ is given by [60]:

$$\hat{\alpha} = h_\alpha^{-1}\left[\hat{R} + \log_2 q\right] \tag{3.26}$$

where R is the estimated entropy rate of z:

$$\hat{R} = -\sum_{z=\ldots,-q,0,q,\ldots} \tilde{p}(z)\log_2 \tilde{p}(z) \tag{3.27}$$

3.2.3 Estimation examples for speech

The results given by the various methods proposed before in order to estimate the shape parameter α of a pdf for a given source are quite similar. "Open-loop" methods are slightly better than "closed-loop" but there complexity is more important. So for practical implementation, we choose to work with a "closed-loop" method and arbitrary we choose Mallat's method [69].

Figure 3.5 shows how the distribution of normalized MDCT coefficients can be approximated by a generalized Gaussian model. Both voiced and unvoiced speech examples are considered. The input signals in time domain (top) are sampled at 16 kHz and transformed by MDCT with a sinusoidal window of 40 ms. The MDCT spectrum of a given frame (middle), comprising 320 coefficients, is normalized by its root mean square (r.m.s.). The histogram of normalized MDCT coefficients is modeled by a generalized Gaussian pdf (bottom). The estimated shape parameter is estimated using Mallat's method [69], for voiced speech $\alpha = 0.29$ and for unvoiced speech $\alpha = 0.53$, so the value of α is somehow related to the voicing of the signal. Note that the value of α would be closer to 2 (Gaussian case) for unvoiced speech if the signals in time domain were linear predictive residuals. However these examples indicate that generalized Gaussian modeling can provide a good approximation of the MDCT spectrum distribution for speech signals. These observations can be extended to music signals.

3.3 Gaussian mixture models (GMMs)

A Gaussian mixture model (GMM) is a weighted sum of Gaussian densities. It is a semi-parametric model determined by the weights, the means and the covariances matrices of the gaussian component [100]. GMM has been developed for vector quantization (VQ) of linear-predictive coding (LPC) parameters [46, 113]. It provides quantization schemes with low complexity and independent from the bitrate.

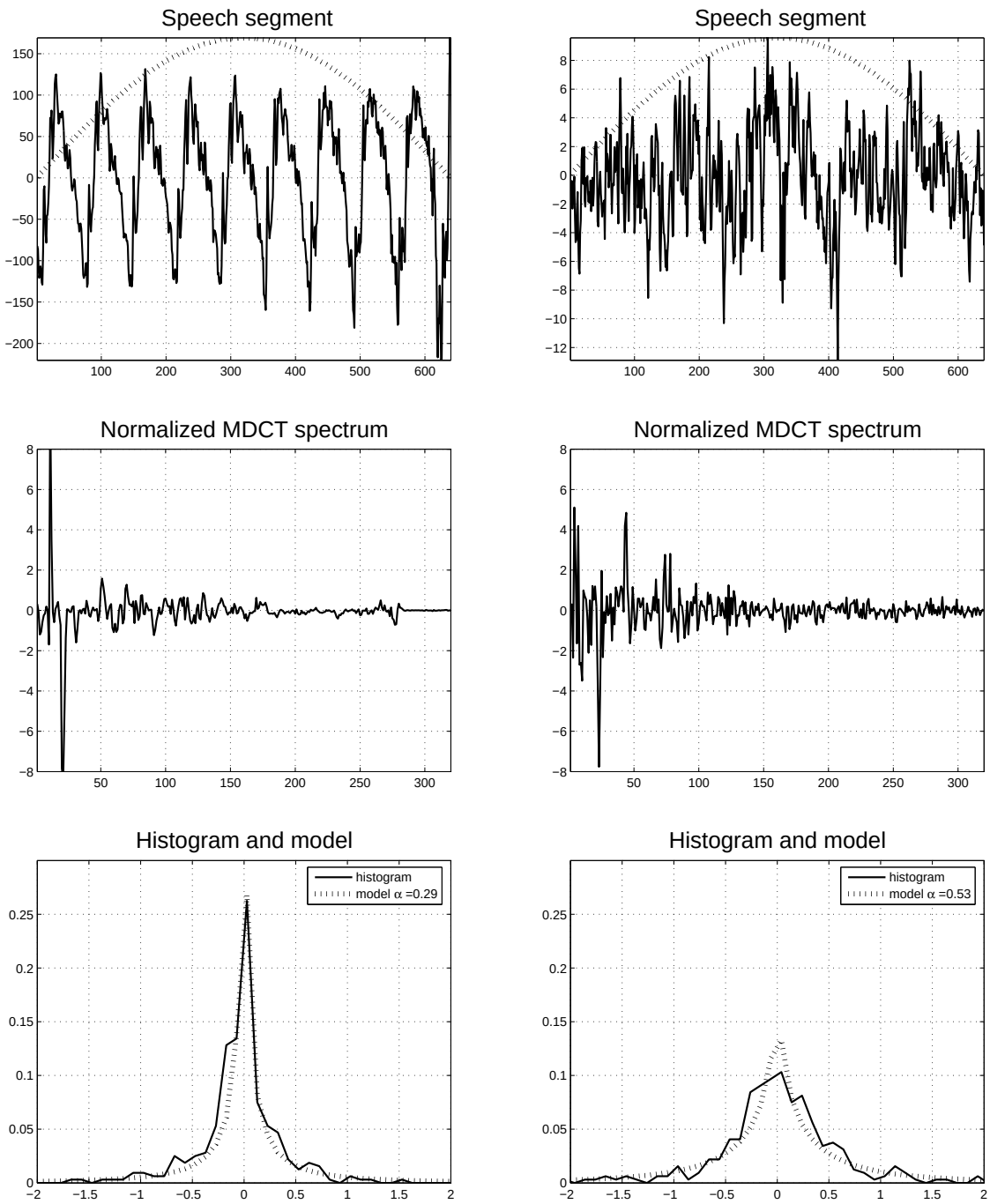
The principle of Gaussian mixture models (GMMs) is given in Section 3.3.1. The expectation-maximization (EM) algorithm which is a practical method to estimate the parameters of the Gaussian mixture is presented in Section 3.3.2.

3.3.1 Principle of Gaussian mixture models

We follow here the notations of [46]. A Gaussian mixture model (GMM) is defined as a weighted sum of Gaussian densities :

$$f(x|\Theta) = \sum_{i=1}^{M} \rho_i \, f_i(x|\theta_i) \,, \tag{3.28}$$

where x, ρ_i and $f_i(x|\theta_i)$ are respectively a vector of dimension n, the mixture weights such as $\sum_{i=1}^{M} \rho_i = 1$ and the Gaussian probability density function of the i-th GMM



(a) Voiced segment (b) Unvoiced segment

Figure 3.5: Examples of MDCT coefficient modeling.

component defined as:

$$f_i(x|\theta_i) = \frac{1}{\sqrt{(2\pi)^n |\det(C_i)|}} \; e^{-\frac{1}{2}(x-\mu_i)^T C_i^{-1}(x-\mu_i)}, \tag{3.29}$$

where μ_i and C_i are respectively the mean vector and the covariance matrix of the i-th GMM component. As a result, a Gaussian mixture model is determined by its parameters:

$$\Theta = \{\rho_1, \ldots, \rho_M, \theta_1, \ldots, \theta_M\} \tag{3.30}$$

where

$$\theta_i = \{\mu_i, C_i\} \tag{3.31}$$

In order to estimate the parameters of each Gaussian in the model the expectation-maximization (EM) algorithm [28, 100] is a practical and commonly used method.

3.3.2 Expectation-maximization algorithm

The EM algorithm is an iterative algorithm to find the maximum-likelihood estimate of the parameters of a distribution from a given data set when the data is incomplete or has missing values [100]. The algorithm guarantees convergence to a locally optimal solution.

In the special case of Gaussian mixture model the incomplete-data log likelihood expression from the data x is given by:

$$log\left(\mathfrak{L}\left(\Theta|x\right)\right) = \sum_{j=1}^{n} log\left(f(x_j|\Theta)\right) = \sum_{j=1}^{n} log\left(\sum_{i=1}^{M} \rho_i\, f_i(x_i|\theta_j)\right) \tag{3.32}$$

which is difficult to optimize because it is a log of the sum. If we consider that the data x are incomplete, and consider the existence of unobserved data items y (whose values inform us which component density "generated" each data item) the log likelihood expression is simplified:

$$log\left(\mathfrak{L}\left(\Theta|x,y\right)\right) = \sum_{j=1}^{n} log\left(\rho_{y_j}\, f_{y_j}(x_j|\theta_{y_j})\right) \tag{3.33}$$

The problem is that we do not know the values of y, so we assume y is a random vector.

Expectation step (E-step)

$\Theta^{opt} = \{\rho_1^{opt}, \ldots, \rho_M^{opt}, \theta_1^{opt}, \ldots, \theta_M^{opt}\}$ are noted as the appropriate parameters for the likelihood $\mathfrak{L}\left(\Theta^{opt}|x,y\right)$. Using Baye's rule we can write the relationship:

$$f\left(y_j|x_j,\Theta^{opt}\right) = \frac{\rho_{y_j}\, f_{y_j}(x_j|\theta_{y_j}^{opt})}{\sum_{i=1}^{M} \rho_i\, f_i(x_j|\theta_i^{opt})} \tag{3.34}$$

We also have the relationship:

$$f\left(y|x,\Theta^{opt}\right) = \prod_{j=1}^{n} f\left(y_j|x_j,\Theta^{opt}\right) \tag{3.35}$$

We try to find the function $\mathcal{Q}\left(\Theta|\Theta^{opt}\right)$ which is defined as the expected value of the complete data log-likelihood with respect to the unknown data given the observation data. After simplification, we have the relationship:

$$\begin{aligned}\mathcal{Q}\left(\Theta|\Theta^{opt}\right) &= \sum_{i=1}^{M}\sum_{j=1}^{n} log\left(\rho_i f_i\left(x_j|\theta_i\right)\right) f\left(i|x_j,\Theta^{opt}\right) \qquad (3.36)\\ &= \sum_{i=1}^{M}\sum_{j=1}^{n} log\left(\rho_i\right) f\left(i|x_j,\Theta^{opt}\right) + \sum_{i=1}^{M}\sum_{j=1}^{n} log\left(f_i\left(x_j|\theta_i\right)\right) f\left(i|x_j,\Theta^{opt}\right)\end{aligned}$$

This expression has to be maximized. It is possible to maximize the term containing ρ_i and the term containing θ_i independently since they are not related.

Maximization step (M-step)

In order to find the expression of ρ_i, we introduce Lagrange multiplier λ under the constraint that $\sum_{i=1}^{M}\rho_i = 1$, and we have the following relationship:

$$\frac{\partial}{\partial\rho_i}\left[\sum_{i=1}^{M}\sum_{j=1}^{n} log\left(\rho_i\right) f\left(i|x_j,\Theta^{opt}\right) + \lambda\left(\sum_{i=1}^{M}\rho_i - 1\right)\right] = 0 \qquad (3.37)$$

So we get:

$$\sum_{j=1}^{n}\frac{1}{\rho_i} f\left(i|x_j,\Theta^{opt}\right) + \lambda = 0 \qquad (3.38)$$

By summing over j, we get $\lambda = -n$ and so we have the following relationship:

$$\rho_i = \frac{1}{n}\sum_{j=1}^{n} f\left(i|x_j,\Theta^{opt}\right) \qquad (3.39)$$

The estimate of the new parameters are as follows:

$$\begin{aligned}\rho_i^{new} &= \frac{1}{n}\sum_{j=1}^{n} f\left(i|x_j,\Theta^{opt}\right) \qquad (3.40)\\ \mu_i^{new} &= \frac{\sum_{j=1}^{n} x_j f\left(i|x_j,\Theta^{opt}\right)}{\sum_{j=1}^{n} f\left(i|x_j,\Theta^{opt}\right)} \qquad (3.41)\\ C_i^{new} &= \frac{\sum_{j=1}^{n} f\left(i|x_j,\Theta^{opt}\right)\left(x_j-\mu_i^{new}\right)\left(x_j-\mu_i^{new}\right)^T}{\sum_{j=1}^{n} f\left(i|x_j,\Theta^{opt}\right)} \qquad (3.42)\end{aligned}$$

3.4 Model-Based Bit Allocation

Model-based bit allocation has been studied in [88] for image coding, we proposed to use it for speech and audio coding. The objectif is to improve the performance of

quantization by having a stepsize which is appropriated for the source. The model used here is the generalized Gaussian model.

The problem of bit allocation is an optimization under constraint. We consider the encoding of N zero-mean random variables $x_1, \ldots, x_N$ of variances σ^2 with respect to the mean square error criterion. We assume that the variables x_i have a generalized Gaussian pdf $g_{\sigma,\alpha}(x)$ of shape parameter α. The variables x_i are coded by scalar quantization with the *same* step size q. We assume that the sequence of integers obtained after scalar quantization is encoded by ideal entropy coding. For a given bit allocation B in bits per sample, the bit allocation problem is to minimize the distortion D under the constraint that $\sum_{i=1}^{N} b_i \leq B$. Solving this problem is the minimization of a function with Lagrangian techniques. The criterion $J(b_i, \lambda)$ is defined as

$$J(b_i, \lambda) = D - \lambda \left(\sum_{i=1}^{N} b_i - B \right) \tag{3.43}$$

where λ is the Lagrange multiplier.

We presented here two cases, one with the hypothesis that we are in high resolution and the other one with the correct formula. This work has been developed in [88].

3.4.1 Asymptotic model-based bit allocation

We follow here the notations of [45] with regards to transform coding and bit allocation. In case of high resolution the mean square error D resulting from the encoding of N random variables x_i is given by [45]:

$$D \approx \sum_{i=1}^{N} h\sigma^2 2^{-2b_i} \tag{3.44}$$

where the constant h_i is a function of the pdf of the variable x_i and b_i is the number of bits per sample used to code x_i. For a generalized Gaussian pdf, h is given by [89]:

$$h = \frac{\Gamma(1/\alpha)^3}{3\alpha^2\Gamma(3/\alpha)} e^{2/\alpha}, \tag{3.45}$$

where α is the shape parameter of the distribution x_i. The distortion D given in Eq. (3.44) can be minimized with Lagrangian techniques. The criterion $J(b_i, \lambda)$ is defined as

$$J(b_i, \lambda) = D - \lambda \left(\sum_{i=1}^{N} b_i - B \right) \tag{3.46}$$

where λ is the Lagrange multiplier.

The derivative of $J(b_i, \lambda)$ with respect to b_i and λ is given by:

$$\frac{\partial J}{\partial b_i} = -2\ln(2)h\sigma^2 2^{-2b_i} + \lambda = 0 \tag{3.47}$$

$$\frac{\partial J}{\partial \lambda} = \sum_{i=1}^{N} b_i - B = 0 \tag{3.48}$$

From Eq. 3.47 we can write:

$$\log_2\left(2\ln(2)h\sigma^2\right) - 2b_i = \log_2(\lambda) \tag{3.49}$$

So the optimal bitrate is given by:

$$b_i = -\frac{1}{2}\log_2(\lambda) + \frac{1}{2}\log_2\left(2\ln(2)h\sigma^2\right) \tag{3.50}$$

We used the relationship obtained in Eq. 3.50 into Eq. 3.48:

$$\sum_{i=1}^{N}\left(\frac{1}{2}\log_2(\lambda) + \frac{1}{2}\log_2\left(2\ln(2)h\sigma^2\right)\right) - B = 0 \tag{3.51}$$

So:

$$-\sum_{i=1}^{N}\log_2\lambda = 2B + \sum_{i=1}^{N}\log_2\frac{1}{2\ln(2)h\sigma^2} \tag{3.52}$$

And finally, we have:

$$\boxed{\lambda = 2^{-2B}2\ln(2)h\sigma^2}. \tag{3.53}$$

The relationship gives the optimal Lagrangian parameter λ_{opt}. With this value of λ the Eq. 3.47 becomes:

$$D = \frac{\lambda_{opt}}{2\ln(2)}. \tag{3.54}$$

Furthermore for high-resolution scalar uniform quantization with step size q, we have [45]:

$$D = \frac{q^2}{12}. \tag{3.55}$$

From (3.54) and (3.55) we find that the optimal stepsize is:

$$\boxed{q = \sqrt{\frac{6\lambda_{opt}}{\ln(2)}}}. \tag{3.56}$$

3.4.2 Non asymptotic model-based bit allocation

The quantization mean square error D_Q resulting for the encoding of N random variables x_i is given by [45]:

$$D_Q = \int_{-z/2}^{z_2} x^2 g_{\sigma,\alpha} dx + 2\sum_{m=1}^{+\infty} \int_{z_2+(m-1)q}^{z_2+mq} \left(x - \hat{x}_m\right)^2 g_{\sigma,\alpha}(x) dx \tag{3.57}$$

where $\hat{x}_m$ is the centroid of each quantization level m defined as :

$$\hat{x}_m = \frac{\int_{-z/2+(m-1)q}^{z/2+mq} x g_{\sigma,\alpha} dx}{\int_{-z/2+(m-1)q}^{z/2+mq} g_{\sigma,\alpha}(x) dx} \tag{3.58}$$

After simplifying we have the following relationship:

$$D_Q = \sigma^2 + 2\sum_{m=1}^{+\infty} \hat{x}_m^2 \int_{-z/2+(m-1)q}^{z/2+mq} g_{\sigma,\alpha}(x) dx - 4\sum_{m=1}^{+\infty} \hat{x}_m \int_{-z/2+(m-1)q}^{z/2+mq} x g_{\sigma,\alpha} dx \tag{3.59}$$

By using Eq. 3.58 we can write that:

$$D_Q = \sigma^2 - 2\sum_{m=1}^{+\infty} \frac{\left(\int_{-z/2+(m-1)q}^{z/2+mq} x g_{\sigma,\alpha}(x) dx\right)^2}{\int_{-z/2+(m-1)q}^{z/2+mq} g_{\sigma,\alpha}(x) dx} \tag{3.60}$$

So the mean square error D is a function of four parameters which are the stepsize q, the deadzone z, the shape parameter α and the variance σ^2. If we consider a function $f_{n,m}\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right)$ defined as:

$$f_{n,m}\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right) = \int_{z/2\sigma+(m-1)q/2\sigma}^{z/2\sigma+mq/2\sigma} x^n g_{1,\alpha}(x) dx\,, \tag{3.61}$$

the Eq. 3.60 is simplified into:

$$D_Q = \sigma^2 \left[1 - 2\sum_{m=1}^{+\infty} \frac{f_{1,m}\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right)}{f_{0,m}\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right)}\right]. \tag{3.62}$$

So we finally have [88]:

$$\boxed{D_Q = \sigma^2 D\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right)}, \tag{3.63}$$

where

$$D\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right) = 1 - 2\sum_{m=1}^{+\infty} \frac{f_{1,m}\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right)}{f_{0,m}\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right)}\,. \tag{3.64}$$

In the special case where the centroid is set to mid-value the distortion D is given by [88]:

$$\begin{aligned} D\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right) &= 1 + 2\sum_{m=1}^{+\infty}\left(\frac{z}{2\sigma} + \left(m - \frac{1}{2}\right)\frac{q}{\sigma}\right)^2 f_{0,m}\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right) \\ &\quad -4\sum_{m=1}^{+\infty}\left(\frac{z}{2\sigma} + \left(m - \frac{1}{2}\right)\frac{q}{\sigma}\right) f_{1,m}\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right) \end{aligned} \tag{3.65}$$

The bit rate is defined as:

$$b = -\sum_{m=-\infty}^{+\infty}(m)\log_2 Pr(m) \tag{3.66}$$

where $Pr(m)$ is the probability of having the quantization level m. The generalized Gaussian distribution is symmetrical so we have the relationship $Pr(m) = Pr(-m)$ and finally:

$$b = Pr(0\log_2 p(0)) - 2\sum_{m=1}^{+\infty} p(m)\log_2 Pr(m) \tag{3.67}$$

where $Pr(m)$ is defined as:

$$Pr(m) = \int_{z/2+(m-1)q}^{z/2+mq/2} g_{1,\alpha}(x)dx = f_{0,m}\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right) \tag{3.68}$$

and $p(0)$ is defined as:

$$p(0) = \int_{-z/2}^{z/2} g_{1,\alpha}(x)dx = f_{0,0}\left(\alpha, \frac{z}{\sigma}\right). \tag{3.69}$$

The bit rate is also a function of four parameters which are the stepsize q, the deadzone z, the shape parameter α and the variance σ^2. The bit rate is defined as [88]:

$$\begin{aligned} b\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right) &= -f_{0,0}\left(\alpha, \frac{z}{\sigma}\right)\log_2 f_{0,0}\left(\alpha, \frac{z}{\sigma}\right) \\ &\quad -2\sum_{m=1}^{+\infty} f_{0,m}\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right)\log_2 f_{0,m}\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right) \end{aligned} \tag{3.70}$$

The solution of the allocation problem can be find by introducing the criterion $J(z, q, \lambda)$ defined as:

$$J(z, q, \lambda) = \left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right) - \lambda\left(b\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right) - B\right) \tag{3.71}$$

where λ is the Lagrange multiplier.

The solution of this equation is presented in [88] and leads to a system of equation which gives charts in order to find the optimal stepsize q under the bit budget constraint.

3.5 Arithmetic Coding

Arithmetic coding is a data compression technique which maps a string of data symbols (codeword) to a code by successive division of the sequence $[0, 1)$ interval (0 is included in the interval and 1 is not included) [62, 126]. Arithmetic coding as other entropy coding techniques requires the knowledge of symbol probabilities [75, 49].

3.5.1 Principles of arithmetic coding

In arithmetic coding, a codeword is represented by an interval of real numbers in $[0, 1)$. Successive symbols of the codeword reduce the size of the interval in accordance with the symbol probabilities and each codeword is the sum of probabilities of the preceding symbols. We considered an alphabet of N symbols $S = \{s_1, \ldots, s_N\}$ of probabilities $P(s_k)$.

Arithmetic coding process [103]

1. Initialization of the first interval $[L, H)$ with $L = 0$ and $H = 1$. The size of this interval is defined as $Size = H - L$.

2. This interval is divided into subintervals $[L_{s_k}, H_{s_k})$ depending of the probability of each symbol s_k of the alphabet. The length of each interval is given by $H_{s_k} - L_{s_k} = P(s_k)$, so:

$$L_{s_k} = L + Size \times \sum_{i=1}^{k-1} P(s_k) \quad \text{and} \quad H_{s_k} = L + Size \times \sum_{i=1}^{k} P(s_k)$$

3. The subinterval corresponding to the next symbol in the codeword is chosen. The initial interval $[L, H)$ is updated:

$$\begin{cases} L = L + Size \times L_{s_k} \\ H = H + Size \times H_{s_k} \end{cases}$$

4. This interval is subdivide again just like in step 2.

5. The steps 2, 3 and 4 are repeated until we get the codeword.

For implementation, arithmetic coding accuracy is limited in order to avoid overflow or underflow in the codeword [103]. Arithmetic coding is slow because of the number of division and multiplication necessary to subdivise an interval. A practical implementation of arithmetic coding is proposed in [126]. It is possible to increase the speed of arithmetic coding without decreasing the performance of compression with approximations likes quasi-arithmetic coding, the patented IBM Q-coder [91, 92], or Neal's low-precision coder [76] .

3.5.2 Example

We considered an alphabet $\{a, e, i, o, u\}$, and a fixed model is used with probabilities shown in Table 3.1. As an example, we considered transmitting the symbols *eiao*, Fig. 3.6 shows the arithmetic coding process for this codeword. The fist symbol is e corresponding to the interval $[0.2, 0.6)$, this interval is subdivided into the same proportions than the original. Thus, the subinterval belonging to the second symbol i is $[0.44, 0.48)$. Then we subdivide $[0.44, 0.48)$ and the subinterval for the third symbol a is $[0.440, 0.448)$. The same process is done for the last symbol o and the subinterval associated is $[0.4456, 0.4472)$.

Table 3.1: Probabilities for a fixed model $\{a, e, i, o, u\}$.

Symbol	Probability	Range
a	0.2	[0, 0.2)
e	0.4	[0.2, 0.6)
i	0.1	[0.6, 0.7)
o	0.2	[0.7, 0.9)
u	0.1	[0.9, 1)

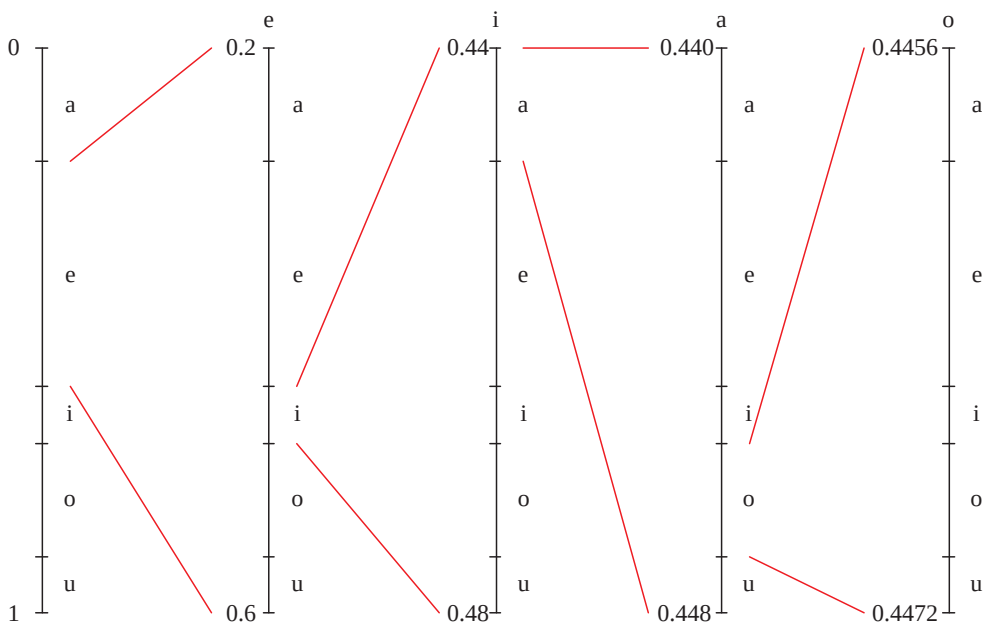


Figure 3.6: Arithmetic coding process.

3.6 Subjective Test Methodology

In order to compare two coders, subjective tests are conduced. The principles of AB tests is presented here. A listener has to compared two coded samples of 8 seconds, the order of the samples is unknown by the listeners. Each comparison is listened two times in order to avoid the order effect: the A sample is in first position (or second) for the first listening and in second position (or first) for the second one. It is possible to listen to the samples the number of times you want and to select only a part of the samples to listen. The order of the listening will always be the first sample first then the second one. After having listening to the samples, the listener has to answer to the question: "What is the difference of quality between the two samples?" and to give a score as in Table 3.2. The obtained scores are comparison mean opinion score (CMOS).

Table 3.2: CMOS for the AB test.

B (-1)	B is better than A
b (-0.5)	B is slightly better than A
= (0)	no preference
a (0.5)	A is slightly better than B
A (1)	A is better than B

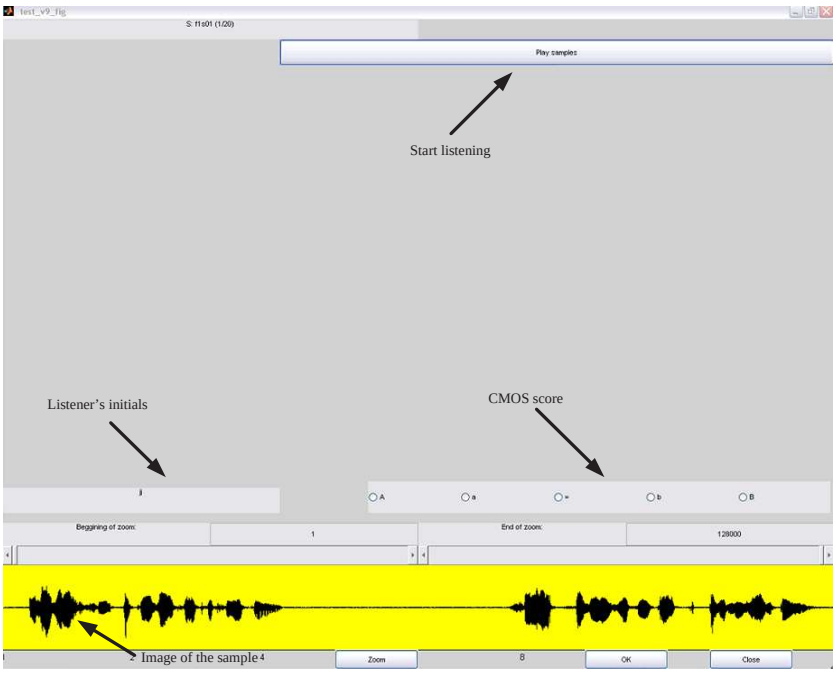


Figure 3.7: Interface of the test.

Fig. 3.7 presents the interface of the listening test. The listener can start the test whenever he wants to and he can listen to a selected part of a sample. The image of the sample does not change from sample a to sample B, so the listener does not have information on the sample is listening.

3.7 Conclusion

In this chapter we presented different methods and models for flexible signal compression. Two statistical models used to approximate the source have been presented: generalized Gaussian model and Gaussian mixture model. GMM will be used to classify linear-predictive coding (LPC) parameters. Principles of model-based bit allocation have also been presented. The model-based bit allocation is used in order to estimate the optimal stepsize and the optimal deadzone for the scalar quantizer.

Chapter 4

Flexible quantization of LPC coefficients

A parametric approach based on Gaussian mixture model (GMM) has been developed for vector quantization (VQ) of linear-predictive coding (LPC) parameters [46, 113]. This approach has brought interest in the design of model-based quantization methods as opposed to standard constrained VQ requiring stochastic codebook training based on a given source database. Several variants of GMM-based LPC quantization have been proposed including predictive methods [113, 80].

In this chapter we present a new model-based coding method to represent the linear-predictive coding (LPC) parameters of wideband speech signals (sampled at 16 kHz). The LPC coefficients are transformed into line spectrum frequencies (LSF) and quantized by switched AR(1)/MA(1) predictive Karhunen-Loeve transform (KLT) coding. KLT coding is equivalent to source coding with singular value decomposition (SVD) or principal component analysis (PCA).

The main contribution of this work lies in the use of improved quantization to represent the (transformed) prediction error of LSF parameters. Generalized Gaussian modeling is applied for this purpose.

This chapter is organized as follows. In Section 4.1 we present shortly the principles of LPC quantization. Previous works on GMM-based LSF vector quantization are presented in Section 4.2, references are given and the GMM-based LSF quantizer of [111] is described. Then in Section 4.3 the proposed LSF vector quantizer is described. This quantizer is based on principal component analysis (PCA) with one Gaussian and the quantizer is build on a generalized Gaussian model. In Section 4.4 the proposed quantizer is modified by introducing a predictive scheme. This prediction can be done by moving-average (MA) or autoregressive (AR) model. Experimental results for predictive KLT coding are presented in Section 4.5 before concluding in Section 4.6.

Part of this work was presented at EUSIPCO'06 [78]

4.1 Background: LPC Quantization

The linear-predictive coding (LPC) parameters are widely used to encode the spectral envelope of speech signal. They described the perceptual important peaks more than the spectral valleys and they are widely used in speech coder to describe the power spectrum envelop. CELP model which is one famous speech coding technique is based on the LPC analysis [106].

LPC analysis

In LPC analysis, an all-pole synthesis filter that models the short-term spectral envelope is used:

$$H(z) = \frac{1}{A(z)} = \frac{1}{1 + \sum_{i=1}^{n} a_i z^{-i}} \tag{4.1}$$

where n and a_i are respectively the model's order and the Linear Prediction Coefficient (LPC) [68, 73]. $A(z)$ is the prediction filter obtained using linear-prediction. In order to have the LPC coefficients a_i, a classical method is to used the Levinson-Durbin algorithm [67]. This method guarantees the poles of the LPC filter to be inside the unit filter and so the resulting filter $H(z)$ is stable.

Scalar quantization of LPC parameters

Direct scalar quantization of LPC parameters is usually not done as small quantization errors can result in large spectral errors and in the instability of the filter $H(z)$. In order to solve this problem the LPC parameters are transformed in representations that ensure stability of $H(z)$. A number of various representations have been proposed such as the reflection coefficients (RC) representation, the arcsine reflection coefficient (ASRC) representation, the log-area ratio (LAR) representation and the line spectral frequency (LSF) representation [86, Chap.12]. The RC, LAR and LSF representations will be presented here.

The reflection coefficients (RC), K_i, can be obtained from the LPC parameters using Levinson-Durbin relationships [67]. The log-area ratio (LAR) coefficients, L_i, are defined as:

$$L_i = \log \frac{1 - K_i}{1 + K_i} \tag{4.2}$$

where K_i is the ith reflection coefficient (RC).

The LSF representation was introduced by [50] and it has a certain number of properties such as sequential ordering of parameters, a simple check of filter stability which makes it a good representation of LPC parameters. To define the LSFs, the inverse filter $A(z)$ is used to construct two polynomials:

$$P(z) = A(z) + z^{n+1} A(z^{-1}) \tag{4.3}$$

$$Q(z) = A(z) - z^{n+1} A(z^{-1}) \tag{4.4}$$

the roots of $P(z)$ and $Q(z)$ are the LSFs.

Possible improvements of LSF quantization

Considerable progress have been made in vector quantization of LPC coefficients in order to reach a good performance/complexity compromise. Improvements can still be expected for the following aspects:

- Computational Complexity: In general, quantizers have considerable search complexity and memory requirements. It is particulary true for vector quantization of LPC parameters with wideband speech signals. So it has lead to practical schemes such as multi-stage or split vector quantizers which use codebooks and sub-optimal search [40]. Despite those sub-optimalities, most schemes have a significant complexity (memory and computational cost).

- Flexibility: Conventional LPC quantization schemes are usually designed for a given bit allocation. So, it is impossible to adapt or modify quantizers "online".

- Learning: Conventional LPC quantization schemes are unsuitable to learning. This is due to the structure of codebooks who do not allow an adaptation online.

4.2 Previous Works on GMM-Based LSF Vector Quantization

The introduction of Gaussian mixture models (GMMs) in quantization was first proposed in [117]. The vector are classified using GMMs. Four methods of quantization are compared and one of them is based on the fact that the coefficients are Gaussian.

Also in image coding, a comparison between quantization of transform coefficients modeled with generalized Gaussian models or GMM has been done [110]. The quantizer used is a DPCM quantizer, and generalized Gaussian is a good model to approximate the distribution of transform coefficients.

In [104], GMMs are used for LPC spectrum coding. GMMs are used to classify LSF vectors for quantization. The coding structure obtain is compared with a predictive vector quantizer (PVQ) and a high rate optimal recursive coder (HRORC). In term of objective and subjective performance the classified VQ is as good as the PVQ but is under the HRORC. Also, a waveform quantization of speech based on Gaussian mixture models is presented in [105]. GMMs are optimized to fit the speech waveform pdf and high dimensional vector quantization using the GMM parameters is used.

In [111], GMMs are exploited in the low complexity design of fixed-rate and variable-rate compression schemes and in the development of joint source-channel decoding architectures. In fixed-rate compression, the pdf of the source is estimated using the GM model and then a low complexity fixed-rate compression scheme based on the model is proposed. A LSF vector quantization is presented in order to illustrate this compression [113]. It has been shown that this quantizer presents good performance compared to a multi-stage quantizer. In [112], two quantization schemes

based on GMMs are presented. One of the model used lattice quantization and the other proposed recursive coding, and both schemes are efficient for narrowband and wideband spectrum quantization.

A flexible entropy-constrained vector quantization scheme based on GMM, lattice quantization and arithmetic coding is proposed in [129]. The computational complexity of the proposed scheme is independent of rate. From theoretical performance is has been shown that, at high rate, the scheme approaches the theoretically optimal performance.

4.2.1 Principle of GMMs-based LSF vector quantization

We present here the GMMs-based LSF vector quantization which has been proposed in [111]. LSF vectors are modeled by GMMs and a vector quantizer is build upon this model. Then in Section 4.2.2 the GMM-based quantizer design is described.

Fig. 4.1 (a) is an example of distribution of GMMs when the order of the mixture is $M = 4$. The LSF vectors are distributed in a superior triangle which explain the position of the Gaussians. In Fig. 4.1 (b) we presents the scheme of GMMs-based LSF vector quantization. The design of the GMM-based quantizer is detailed in Section 4.2.2. An input LSF vector $\mathbf{x}$ of dimension n is coded M times. If we have a mixture of 8 Gaussians, the input LSF vector $\mathbf{x}$ will be coded 8 times. Then the quantized LSF vector $\hat{\mathbf{x}}$ is selected among M candidates $\widehat{\mathbf{x}}^{(i)}$, with $i = 1, \cdots, M$, by minimizing a distortion criterion as illustrated in Fig. 4.1 (b):

$$\hat{\mathbf{x}} = \hat{\mathbf{x}}^{(j)} \text{ where } j = \arg \min_{i=1,\cdots,M} d(\mathbf{x}, \widehat{\mathbf{x}}^{(i)}). \tag{4.5}$$

The selection criterion d can be the log-spectral distortion [113] or a simple weighted Euclidean distance [99].

4.2.2 GMM-based quantizer design

Fig. 4.2 presents the design of the quantizer based on GMMs for an input LSF vector $\mathbf{x}$ of dimension n. Firstly, GMMs are designed on a training database extract from the source $\mathbf{x}$. A set of parameters which defines the GMMs (see also Section 3.3) is extracted:

$$\Theta = \{\rho_1, \ldots, \rho_M, \theta_1, \ldots, \theta_M\} \tag{4.6}$$

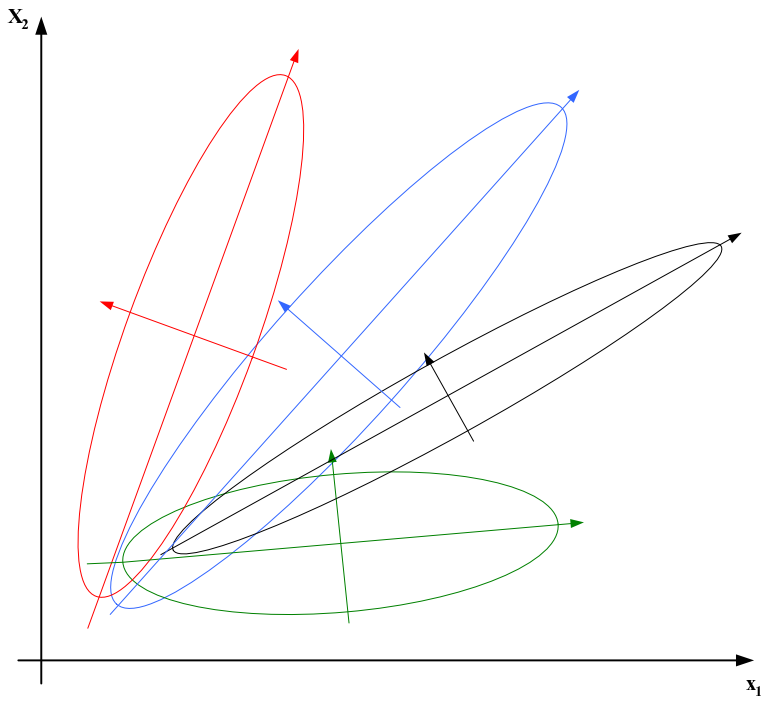
where

$$\theta_i = \{\mu_i, C_i\} \tag{4.7}$$

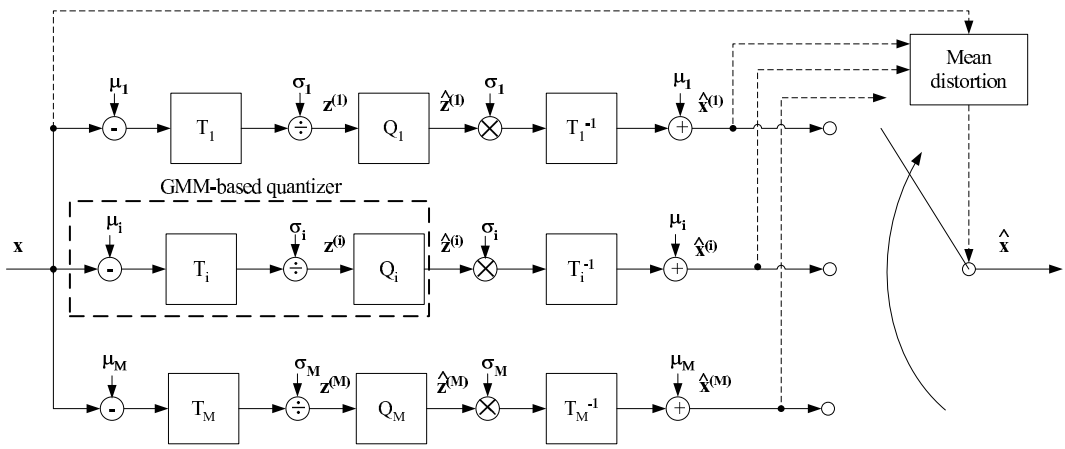
where ρ_i, μ_i and C_i are respectively the mixture weights, the mean vector and the covariance matrix of the i-th GMM component.

The source is then computed by mean-removed Karhunen-Loeve transform (KLT) coding using the parameters μ_i and C_i of the i-th GMM component (see [113]). The KLT matrix T_i and normalization factors $\sigma_i = (\sigma_{i1}, \ldots, \sigma_{in})$ are derived from the eigenvalue decomposition of covariance matrices C_i [113]:

$$C_i = T_i \ diag(\sigma_{i1}^2, \cdots, \sigma_{in}^2) \ T_i^T \tag{4.8}$$



(a) Example of distribution of GMM with 4 components.



(b) GMMs-based LSF vector quantization scheme.

Figure 4.1: Principle of GMM-based LSF vector quantization.

where $\sigma_{i1}^2 \geq \ldots \geq \sigma_{in}^2$ are the eigenvalues of C_i and the matrix T_i comprises the eigenvectors of C_i. A normalization by σ_i is applied on the signal afterwards. The obtained signal $\mathbf{z}^{(i)}$ is then defined as:

$$\mathbf{z}^{(i)} = \frac{1}{\sigma_i} \times T_i(\mathbf{x} - \mu_i) \tag{4.9}$$

The signal $\mathbf{z}^{(i)}$ is then scalar quantized and the obtained signal is $\hat{\mathbf{z}}^{(i)}$. The quantization can be done using a companded scalar quantizer which is presented in

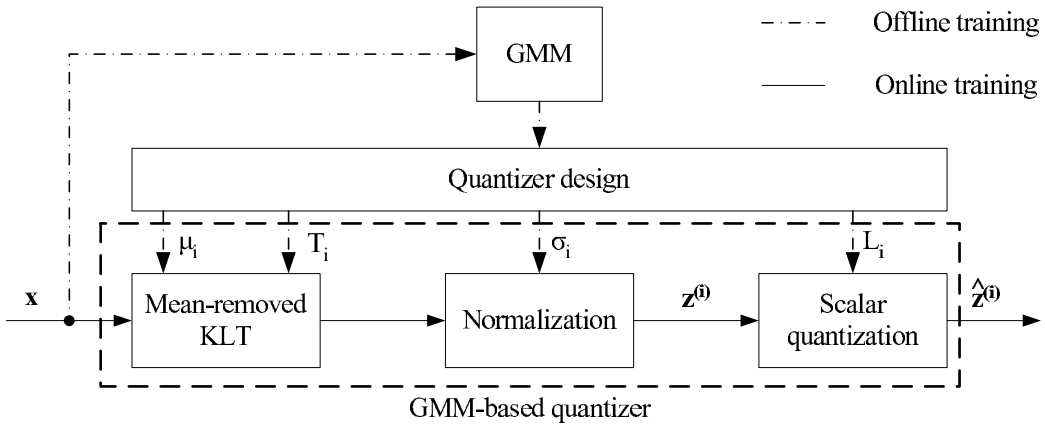


Figure 4.2: Quantizer design based on GMMs.

the next paragraph [113]. In order to have quantization levels $L_i = (L_{i1}, \ldots, L_{in})$ a bit allocation algorithm is implemented and presented in the followings.

To simplify the notation we denote source $\mathbf{z}^{(i)}$ by $\mathbf{z} = (z_1, \ldots, z_j, \ldots, z_n)$ where z_j is the j-th component. Also we denote scalar quantization level L_{ij} by L_j.

Companded scalar quantization of Gaussian variables

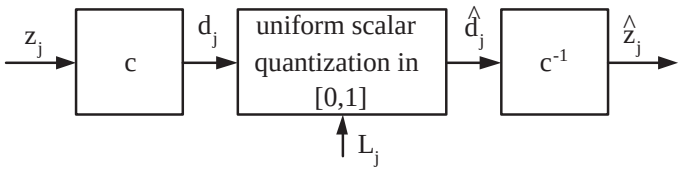


Figure 4.3: Coding of uncorrelated Gaussian components by companded VQ.

As shown in Fig. 4.3 the source $\mathbf{z} = (z_1, \ldots, z_n)$ can be encoded by companded scalar quantization. Under high-rate and Gaussian assumptions the optimal companding for a unit-variance random variable z is given by [80]:

$$c(z) = \frac{1}{2}(1 + erf(\mathbf{z}/\sqrt{6})), \tag{4.10}$$

where erf is the error function:

$$erf(\mathbf{z}) = \frac{2}{\sqrt{\pi}} \int_0^z e^{-t^2} \; dt. \tag{4.11}$$

The inverse operation is given by:

$$c^{-1}(\mathbf{z}) = \sqrt{6} \;\; erf^{-1}(2\mathbf{z} - 1). \tag{4.12}$$

The source $\mathbf{z} \in [-\infty, +\infty]^n$ is mapped into a source $\mathbf{d} \in [0,1]^n$ with $\mathbf{d} = (c(z_1), \cdots, c(z_N))$. Uniform scalar quantization in $[0,1]^n$ is applied to $\mathbf{d}$. If d_j is quantized with $L_j \geq 1$ scalar levels, the reconstruction $\hat{d}_j$ is given by

$$\hat{d}_j = ([d_j L_j - \frac{1}{2}] + \frac{1}{2})/L_j \tag{4.13}$$

where [.] denotes the rounding to the nearest integer.

Bit allocation between each Gaussian component of the mixture

For a bit budget B per vector, the number of bits b_i allocated to the i-th GMM component is constrained so that $2^{b_1} + \cdots + 2^{b_M} = 2^B$. In case of high resolution the mean square distortion D for a n-dimensional case [45] is given by :

$$D = \sum_{i=1}^{M} \rho_i \frac{\sqrt{3}\pi}{2} n \Sigma_i 2^{-2b_i/n} \tag{4.14}$$

where ρ_i and Σ_i are respectively the mixture weights, and a function of the eigenvalue decomposition of covariance matrices C_i. Σ_i is defined as:

$$\Sigma_i = \left[\prod_{j=1}^{n} \sigma_{ij}^2 \right]^{\frac{1}{n}} \tag{4.15}$$

where σ_{ij}^2 are normalized factors derived from the eigenvalue decomposition of covariance matrices C_i.

The mean square distortion D under the constraint $\sum_{i=1}^{M} 2^{b_i} = 2^B$ can be minimized by Lagrangian techniques:

$$\begin{aligned} L(b_i) &= D - \lambda \left(\sum_{i=1}^{M} 2^{b_i} - B \right) && (4.16) \\ &= \sum_{i=1}^{M} \rho_i \frac{\sqrt{3}\pi}{2} n \Sigma_i 2^{-2b_i/n} - \lambda \left(\sum_{i=1}^{M} 2^{b_i} - B \right) && (4.17) \end{aligned}$$

We can derive L with respect to b_i and λ and we get the following equations for $1 \leq i \leq M$:

$$\begin{aligned} \frac{\partial L}{\partial b_i} &= \left(\sqrt{3}\pi \rho_i \Sigma_i \right)^{n/(n+2)} - 2^{b_i} \lambda^{n/(n+2)} = 0 && (4.18) \\ \frac{\partial L}{\partial \lambda} &= \sum_{i=1}^{M} 2^{b_i} - 2^B = 0 && (4.19) \end{aligned}$$

From Eq. 4.18 we can write that:

$$\boxed{\lambda^{n/(n+2)} = \left(\sqrt{3}\pi \rho_i \Sigma_i \right)^{n/(n+2)} \times 2^{-b_i}} \tag{4.20}$$

So we have the following relationship:

$$2^{b_i} = \left(\frac{\sqrt{3}\pi\rho_i\Sigma_i}{\lambda} \right)^{n/(n+2)} \tag{4.21}$$

We are summing Eq. 4.21 on i so:

$$\sum_{i=1}^{M} 2^{b_i} = 2^B = \sum_{i=1}^{M} \left(\frac{\sqrt{3}\pi\rho_i\Sigma_i}{\lambda} \right)^{n/(n+2)} \tag{4.22}$$

$$= \frac{\sqrt{3}\pi}{\lambda^{n/(n+2)}} \times \sum_{i=1}^{M} (\rho_i\Sigma_i)^{n/(n+2)} \tag{4.23}$$

$$= \frac{2^{b_i}}{(\rho_i\Sigma_i)^{n/(n+2)}} \times \sum_{i=1}^{M} (\rho_i\Sigma_i)^{n/(n+2)} \tag{4.24}$$

Finally we have the relationship between the number of bits b_i allocated to the i-th GMM component and the bit budget B given as:

$$\boxed{2^{b_i} = 2^B \frac{(\rho_i\Sigma_i)^{n/(n+2)}}{\sum_{i=1}^{M} (\rho_i\Sigma_i)^{n/(n+2)}}} \tag{4.25}$$

Bit allocation between each KLT coefficients

The bit allocation between each quantization level L_j is done by a greedy algorithm [45]. Given a bit budget b_i and the covariance matrix $diag(\sigma_1^2, \cdots, \sigma_n^2)$ the allocation procedure consists of the following steps:

1. Initialize: $L_j = 1$, $j = 1, \cdots, n$.
2. While $\prod_{j=1}^{n} L_j \leq 2^{b_i}$, $L_k := L_k + 1$ where $k = \arg\max_{j=1,\cdots,n} (\sigma_j/m_j)^2$
3. While $\prod_{j=1}^{n} L_j \leq 2^{b_i}$, $L_k := L_k + 1$ where k tests all positions from 1 to n sorted according to $(\sigma_j/m_j)^2$, $j = 1, \cdots, n$.

The principle of this algorithm is to give bits to the most needed quantizer. The initialization of the quantization level L_j can take into account the distribution of the coefficients.

4.3 Proposed Model-Based LSF Vector Quantization

In order to improve the performance of the GMM-based LSF VQ in terms of complexity , flexibility and learning, we proposed to include a model-based quantizer.

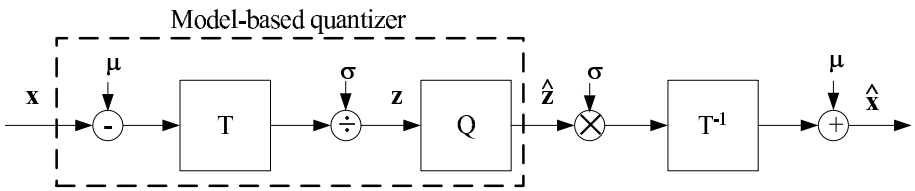


Figure 4.4: Model-based LSF vector quantization.

The pdf of $\mathbf{z}$ is approximated by a generalized Gaussian model presented in Section 3.2 and the quantizer is based on this model.

Fig. 4.4 presents the proposed model-based LSF quantizer design. The design of the model-based quantizer is presented in Section 4.3.1. We restricted ourselves to one Gaussian, the motivations for such restriction are as follows:

- The complexity of GMM-based VQ is roughly linear in the GMM order. This is true for both storage requirement and computational cost. For instance, typical LSF quantization methods [113, 80] use a GMM of 4 or 8, which implies a significant complexity overhead compared to KLT coding with one Gaussian component.

- The bit mapping in GMM-based VQ is usually not optimized to be robust against bit errors. The related bit allocation methods [113] distribute a certain amount of codewords among Gaussian components, in such a way that a single bit error in the overall quantization index can have a dramatic impact on the reconstructed LSF parameters. KLT coding with one Gaussian component can avoid this problem.

4.3.1 Model-based quantizer design

Fig. 4.5 presents the model-based quantizer design. An input LSF vector $\mathbf{x}$ is coded by principal component analysis. A training database extracted from vectors $\mathbf{x}$ is used in order to estimate the Gaussian defined by the followings parameters:

$$\theta = \{\mu, C\} \tag{4.26}$$

where μ, C are respectively the mean vector and the covariance matrix of the Gaussian. The source $\mathbf{x}$ is then computed by mean-removed KLT as in Section 4.2.2 and a normalization by $\sigma = (\sigma_1, \ldots, \sigma_n)$, where σ_j are the eigenvalues of covariance matrices C, is applied after.

A training database is extracted from vectors $\mathbf{z}$ in order to estimate the pdf of each component z_j by a generalized Gaussian model $g_{\alpha_j,\sigma_{gj}}$, where α_j and σ_{gj} are respectively the shape parameters and the standard deviation of the generalized Gaussian model (see also Section 3.2). The vector $\mathbf{z}$ is normalized so the standard deviation $\sigma_{gj} = 1$ and it is not necessary to transmit it. Vector $\alpha = (\alpha_1, \ldots, \alpha_n)$ of shape parameters is then transmitted to the quantizer design. The vector $\hat{\mathbf{z}}$ is

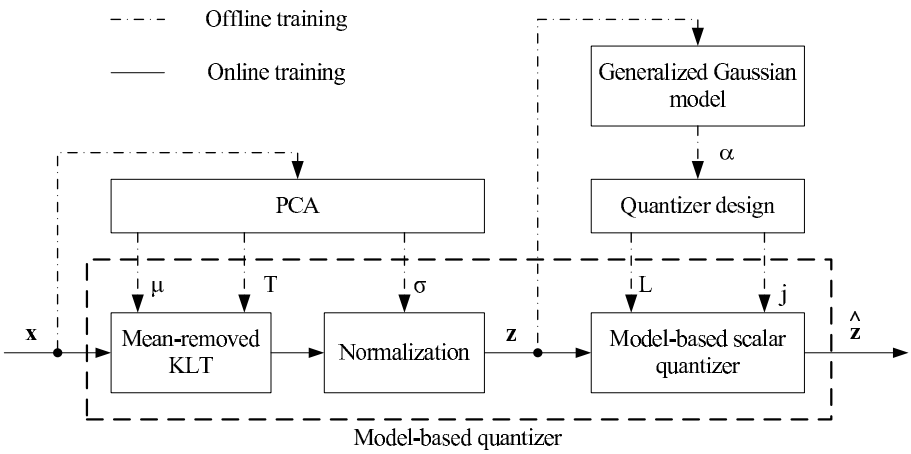


Figure 4.5: Model-based LSF quantization design.

obtained after model-based quantization of $\mathbf{z}$. The quantizer is a model-based Lloyd-Max quantizer presented in Section 4.3.2. The initialization of quantization levels $L = (L_1, \ldots, L_n)$ for the greedy algorithm (see Section 4.2.2) is presented in Section 4.3.3.

4.3.2 Lloyd-Max quantization for a generalized Gaussian pdf

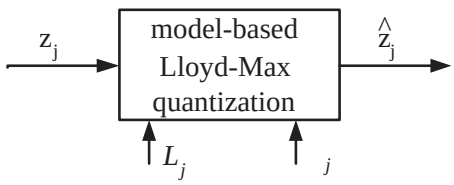


Figure 4.6: Coding of uncorrelated Gaussian components by Lloyd-Max quantizer.

In Figure 4.6, the source $\mathbf{z} = (z_1, \cdots, z_n)$ is encoded by Lloyd-Max quantization [53]. We optimize here the Lloyd-Max quantizer using the theoretical pdf of the source model. This model-based approach allows to circumvent the costly training of stochastic codebooks using a database. This makes the quantization more versatile, as it can be easily reoptimized by updating the generalized Gaussian model parameters. This "theoretical" Lloyd-Max quantizer does not have an important computational complexity.

For an initial codebook $\{s_1, \ldots, s_{L_j}\}$, the decision thresholds t_i, and the recon-

struction levels s_i of the quantizer are found by the following iterative process :

$$t_i = \frac{1}{2}(s_i + s_{i-1}) \qquad i = 2, \ldots, L_j \tag{4.27}$$

$$\text{with } t_0 = -\infty \text{ and } t_{L_j+1} = +\infty$$

$$s_i = \frac{\int_{t_i}^{t_{i+1}} z \; g_{\alpha_j,1}(z)\, dz}{\int_{t_i}^{t_{i+1}} g_{\alpha_j,1}(z)\, dz} \qquad i = 1, \ldots, L_j \tag{4.28}$$

where L_j is the allocated number of quantization levels and $g_{\alpha_j,1}$ is the pdf of the generalized Gaussian model of shape parameters α_j.

4.3.3 Initialization of bit allocation for KLT coefficients with a generalized Gaussian model

Instead of initialize quantization levels L_j to 1 in the greedy algorithm, a method based on the generalized Gaussian model is proposed here.

The problem of bit allocation to several generalized Gaussian random variables has been studied in [88, 89]. In case of high resolution, for a given allocation per sample $R_{tot} = \frac{1}{n}\sum_{j=1}^{n} R_j$, the scalar level quantization $L_{j=1\ldots n}$ for a random variable $z_{j=1\ldots n}$ of shape parameter $\alpha_{j=1\ldots n}$ is defined as (see also Section 3.4):

$$\boxed{L_j = 2^{R_j} = 2^{-\frac{1}{2}\log_2\left(\frac{\lambda}{n}\right)+\frac{1}{2}\log_2(2\log(2)\mathcal{F}(\alpha_j))}}, \tag{4.29}$$

$$\text{where} \quad \mathcal{F}(\alpha_j) = \frac{\Gamma(1/\alpha_j)^3}{3\alpha_j^2\Gamma(3/\alpha_j)} e^{\frac{2}{\alpha_j}} \tag{4.30}$$

$$\text{and} \quad \lambda = 2^{-R_{tot}} 2\log(2) \prod_{i=1}^{n} \left(\frac{\mathcal{F}(\alpha_j)}{n}\right). \tag{4.31}$$

4.4 Inclusion of a prediction

The model-based quantizer presented in Section 4.3 is modified in order to include a prediction. Fig. 4.7 presents the inclusion of the prediction into the model-based quantizer. Two prediction models are proposed: autoregressive (AR) predictive coding and moving-average (MA) predictive coding.

In our quantizer we proposed to have a switch between the two models as in Fig. 4.8. The input LSF vector $\mathbf{x}$ is quantized respectively by the two models and the quantized LSF vector $\hat{\mathbf{x}}$ is selected among the two candidates by minimizing a distortion criterion as presented in Section 4.2.1.

4.4.1 AR predictive coding

Autoregressive (AR) predictive model is a closed-loop predictive quantizer, widely known as differential pulse code modulation (DPCM). This technique is common in audio and image compression standard.

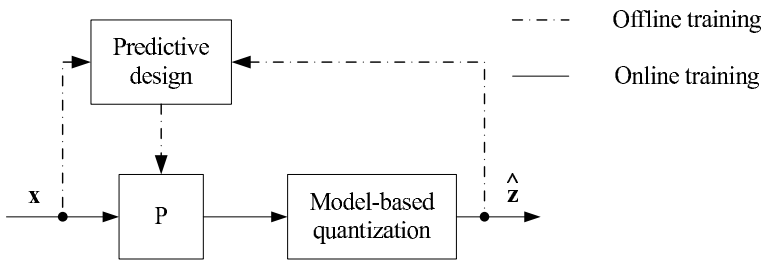


Figure 4.7: Inclusion of a prediction in the model-based quantizer.

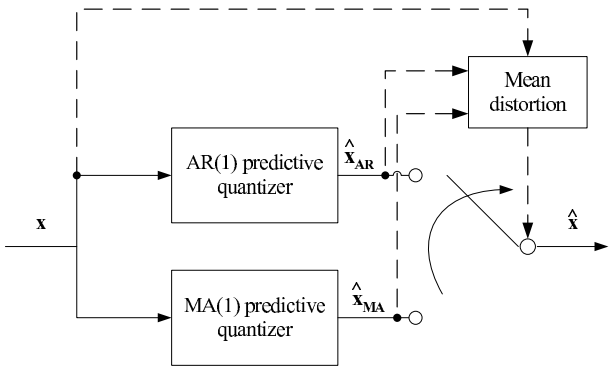


Figure 4.8: Switch between the two predictive models.

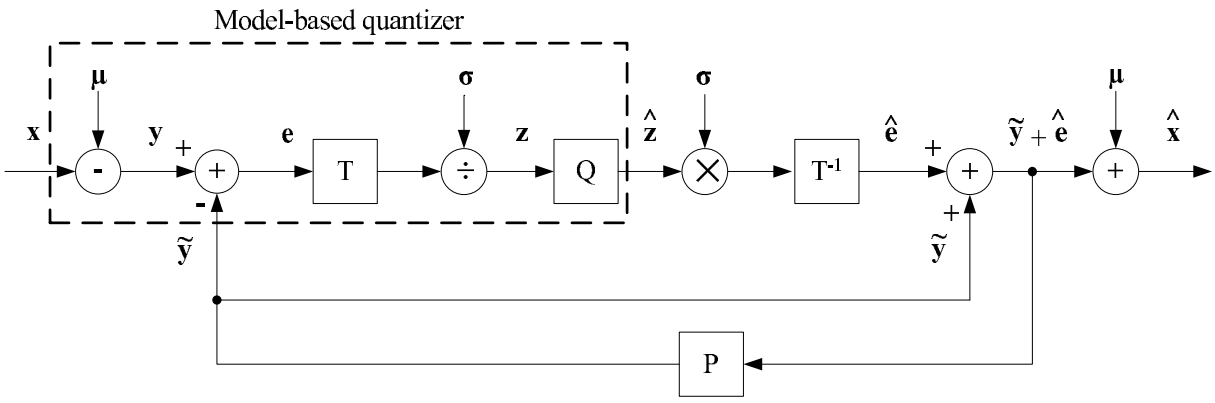


Figure 4.9: Principle of model-based VQ with AR predictive coding.

Autoregressive (AR) predictive model-based coding is illustrated in Fig. 4.9. In this case we define the mean-removed input LSF vector $\mathbf{y}$ of dimension n where $\mathbf{y} = \mathbf{x} - \mu$ and μ is the long-term mean of $\mathbf{x}$. A Gaussian approximates the pdf of the prediction error $\mathbf{e} = \mathbf{y} - \tilde{\mathbf{y}}$. An autoregressive (AR) linear predictor P is used to compute the prediction $\tilde{\mathbf{y}}$. The actual quantization of $\mathbf{e}$ follows the model-based LSF VQ described in Section 4.3. For an autoregressive predictor of order 1, the

prediction $\tilde{\mathbf{y}}$ is defined as:

$$\tilde{\mathbf{y}} = p\,(\hat{\mathbf{e}} + \tilde{\mathbf{y}}) \tag{4.32}$$

where p is the AR(1) prediction matrix. It has been shown in [45, Chap.7][22, 44] that the closed-loop prediction gain ratio is given by:

$$G = \frac{\sigma_{\mathbf{x}}^2}{\sigma_{\mathbf{e}}^2} \tag{4.33}$$

where $\sigma_{\mathbf{x}}^2$ and $\sigma_{\mathbf{e}}^2$ are respectively the variance of $\mathbf{x}$ and $\mathbf{e}$. In order to initialize the prediction matrix we choose $p = G$. This coefficient is updated for each frame and it converge after a short number of iterations.

4.4.2 MA predictive coding

Moving-average (MA) predictive model is an open loop predictive quantizer. This technique present the advantage of reducing quantization error compared to AR predictive model.

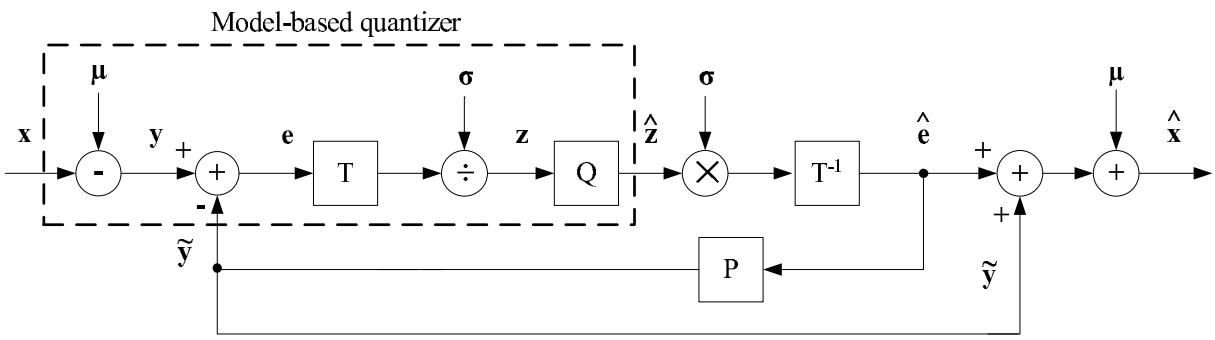


Figure 4.10: Principle of model-based VQ with opened-loop predictive coding.

Moving-average (MA) predictive model-based coding is illustrated in Fig. 4.10. The principle is the same than for AR predictive model-based coding presented in Section 4.4.1. A moving-average linear predictor P is used to compute the prediction $\tilde{\mathbf{y}}$. For a moving-average predictor of order 1, the prediction $\tilde{\mathbf{y}}$ is defined as:

$$\tilde{\mathbf{y}} = p\hat{\mathbf{e}} \tag{4.34}$$

where p is the MA(1) prediction matrix. In our case p is the prediction matrix used in AMR-WB LPC quantization [40] defined as:

$$p = diag(1/3, \cdots, 1/3) \tag{4.35}$$

4.5 Experimental results for predictive model-based LPC quantization

The database used for this work is the NTT-AT wideband speech material (sampled at 16 kHz) which is multilingual, multi-speaker and lasts 5 hours – this material is stored on four CDs. The downsampling to 12.8 kHz and linear-predictive analysis of AMR-WB [12] is used to extract LSF vectors of dimension 16. Note that silence frames are discarded. Three CDs are selected to build a training database comprising 607,386 LSF vectors, while the other CD is used to generate a test database of 202,112 LSF vectors.

For the predictive KLT coding,the E-M algorithm [28] is applied to the training database to estimate the GMM parameters for an order M =1 (with full covariance matrices, and means initialized by the generalized Lloyd-Max algorithm).

4.5.1 Shape parameters of generalized Gaussian models

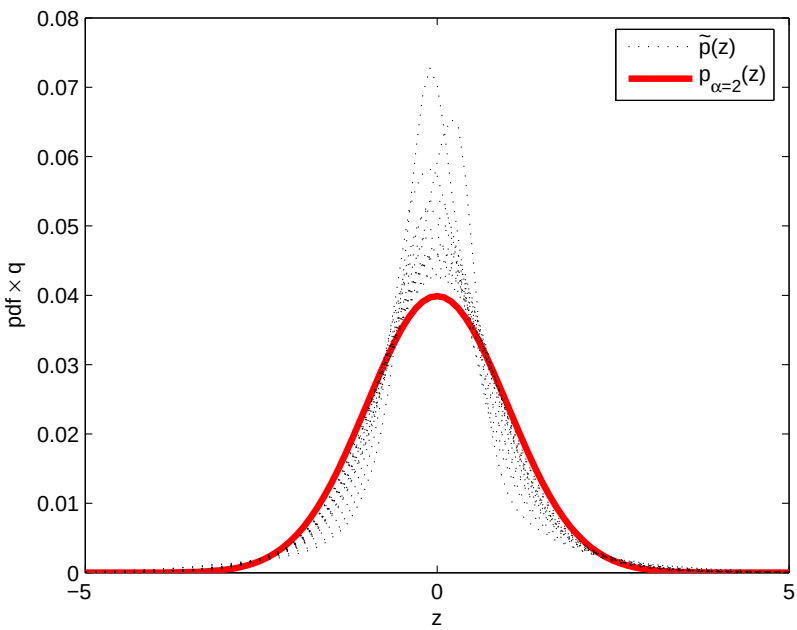


Figure 4.11: Normalized histograms $\tilde{p}(z_j)$ of data samples $z_{j=1...16}$ compared with a Gaussian model with $p_{\alpha=2}(z) = g_{\alpha=2}(z) \times q$.

Figure 4.11 compares the normalized histograms $\tilde{p}(z_j)$ of samples $z_{j=1...16}$ with a Gaussian distribution, while Table 4.1 presents the values of the shape parameter α_j for the estimation methods using the same step size q. These results show that generalized Gaussian modeling provides a better approximation than a Gaussian

model. Furthermore, it turns out that methods in "closed loop" give better results than the "open loop" ones. For the next part, the estimation method of the shape parameter will use the method proposed by Mallat. This allows to optimize design parameters in a very efficient way. The shape parameters $\alpha_{j=1...16}$ are between 0.89 and 1.70 with the method proposed by Mallat.

Table 4.1: Shape parameters $\alpha_{j=1...16}$ estimated on the training database of vectors $\mathbf{z}$.

j	χ^2	KS	KL	Mallat	κ_α	h_α
1	1.04	1.11	1.03	0.98	0.93	0.91
2	1.14	1.31	1.14	1.11	1.06	1.05
3	1.15	1.22	1.15	1.12	1.05	1.08
4	1.18	1.28	1.17	1.14	1.05	1.09
5	1.19	1.16	1.19	1.15	1.06	1.11
6	1.35	1.42	1.35	1.32	1.20	1.24
7	1.39	1.32	1.39	1.37	1.26	1.31
8	1.44	1.44	1.43	1.40	1.25	1.32
9	1.50	1.50	1.50	1.49	1.37	1.42
10	1.48	1.52	1.47	1.46	1.33	1.39
11	1.47	1.46	1.47	1.46	1.33	1.38
12	1.52	1.54	1.52	1.51	1.40	1.45
13	1.63	1.63	1.62	1.62	1.49	1.54
14	1.66	1.68	1.66	1.65	1.51	1.57
15	1.71	1.73	1.71	1.70	1.56	1.61
16	0.95	0.99	0.94	0.89	0.86	0.87

4.5.2 Spectral distortion statistics

The performance of LSF quantization is evaluated with the usual spectral distortion (SD) [86, Chap.12]. The SD statistics obtained for switch AR(1)/MA(1) predictive LSF quantization can be found in Table 4.2. The total bit allocation is 36 or 46 bits; one bit is used to indicate the predictor switch, while the rest – 35 or 45 bits – is allocated to quantize the transformed prediction error $\mathbf{z}$.

The results show that model-based Lloyd-Max (LM) quantization improves the performance compared to companded scalar quantization (SQ). This is due to the fact that the related companding is optimized for a Gaussian source coded at high bit rates; yet, the high-bit rate assumption is not valid for practical LPC quantization. In the case of LM quantization with $\alpha = 2$ the gain in average SD over companded SQ is around 0.03-0.11 dB but the amount of outliers is slightly increased. In the case of LM quantization with optimal α the gain in average SD is more significant (around 0.05-0.13 dB) and the amount of outliers is significantly reduced.

Table 4.2: Results for switched AR(1)/MA(1) predictive LSF quantization vs AMR-WB LPC quantization: comparison between companded SQ and model-based LM quantization.

(a) Results at 36 bits per frame

Quantization methods	avg. SD (dB)	$SD \geq 2$ dB (%)	$SD \geq 4$ dB (%)
Companded SQ	1.36	6.71	0.675
LM, $\alpha = 2$	1.25	9.43	1.480
LM, optimal α	**1.23**	**6.51**	**0.293**
AMR-WB [40]	1.13	3.01	0.015

(b) Results at 46 bits per frame

Quantization methods	avg. SD (dB)	$SD \geq 2$ dB (%)	$SD \geq 4$ dB (%)
Companded SQ	0.90	2.42	0.323
LM, $\alpha = 2$	0.87	4.20	0.710
LM, optimal α	**0.85**	**1.71**	**0.057**
AMR-WB [40]	0.78	0.45	0.003

If we compare the performance of Lloyd-Max quantization for $\alpha = 2$ and optimal α, it turns out that generalized Gaussian modeling still brings a non-neglegible improvement.

The performance of the LPC quantizer used in AMR-WB is also reported. It shows that the performance of the proposed model-based coding is close (but slightly inferior to) classical constrained VQ. The performance gap between AMR-WB and LM with optimized α is around 0.07-0.10 dB in average SD.

4.5.3 Complexity

The memory requirement for fixed-point AMR-WB LPC quantization tables is 6.7 kword (16-bit words) [40]. For the predictive KLT coding with model-based Lloyd-Max quantization, the memory requirement is 0.8 kword assuming a fixed-point implementation. In particular we need to store for each bit allocation (36 or 46 bits) the KLT matrix T, the eigenvalues σ_j, the LM centroids as well and the number of quantizations. It would be possible to reduce even more the memory consumption, for instance by forcing a unique KLT matrix T independent of bit allocation. The memory cost of model-based Lloyd-Max quantization is very small, yet it depends slightly on bit allocation. On the other hand non-uniform scalar quantization with high-rate Gaussian companding has a complexity independent of bit allocation, yet this technique implies to store tables to implement the compander c.

Moreover, the total computational cost of predictive KLT coding with model-based Lloyd-Max quantization in fixed-point is estimated around 0.1 wMOPS (weighted Million Operations per Second) whereas it is around 1.9 wMOPS for

AMR-WB LPC quantization.

4.6 Conclusion

In this chapter we presented a predictive KLT quantization method using generalized Gaussian modeling for wideband LSF speech parameters. This method was compared to AMR-WB LPC quantization [40] and to GMM-based LSF quantizer presented by [113]. The proposed method has much lower complexity (computation cost, storage requirement) and similar performance in term of average SD than AMR-WB LPC quantization. The number of outliers in the proposed method is more important than AMR-WB quantization (2 times at 36 bits per frame and 4 times at 46 bits per frame), but the percentage is too small to affect the quality at 46 bits per frame and it might affect it a little at 36 bits per frame [86, Chap.12]. The pdf knowledge of (transformed) prediction error of LSF parameters improves the performance (average SD and percentage of outliers) compared to GMM-based LSF quantizer and the complexity is equivalent. We have presented a model-based predictive quantizer method using the generalized Gaussian model which performs as well as standard coding methods and improves GMM-based coding. This technique has been applied for LSF parameters but could be used for any signals.

Chapter 5

Flexible Multirate Transform Coding based on Stack-Run Coding

Many efforts have been focused on improving audio quality in telecommunication services by moving from narrowband speech coding (300-3400 Hz) to wideband speech coding (50-7000 Hz) [97, 52]. Besides extending audio bandwidth, quality can be improved by optimizing the quantization of existing coding models.

We present a new model-based method to code transform coefficients of audio signals for audio and video conferencing applications. We compare ourselves with ITU-T G.722.1 [39] which is an example of wideband speech and audio coding system that is used in audio and video conferencing applications. G.722.1 is built upon modified discrete cosine transform (MDCT), scalar quantization and vector Huffman coding of normalized MDCT coefficients. We propose in this work a different coding method for MDCT coefficients with the objective to improve coding efficiency.

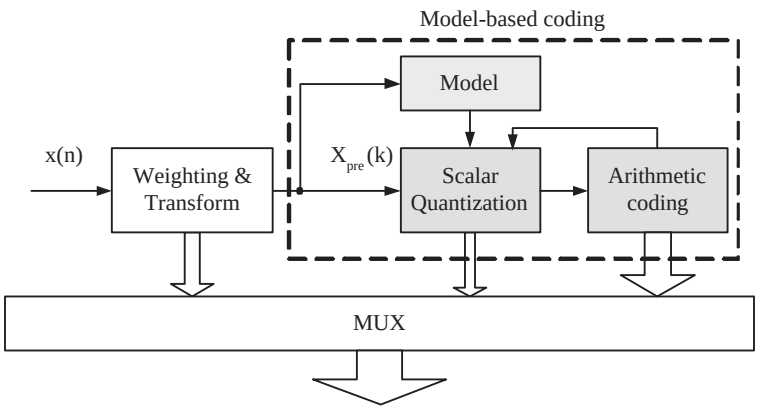


Figure 5.1: Proposed model-based transform coder.

Fig. 5.1 presents the principle of the proposed model-based transform audio coding. The box in grey is the novelty compared to a standard transform audio

Part of this work was presented at the 120th AES Convention [77] and ICASSP'07 [79]

coding. An input signal $x(n)$ is mapped into a "perceptual" domain by weighting and transform operations described in Section 5.2. The distribution of pre-shaped coefficients X_prek is approximated by a statistical model. Here we are using generalized Gaussian modeling as described in Section 3.2. Then a scalar quantization based on the model is applied on the coefficients $X_{pre}(k)$ and finally they are coded by arithmetic coding, here we are using stack-run coding.

The main contribution of this work lies in the use of arithmetic-coded scalar quantization and the application of generalized Gaussian modeling for efficient bit allocation and deadzone optimization. Generalized Gaussian modeling is commonly used in image and video coding [89] but its application to speech and audio coding is quite new. Using generalized Gaussian modeling provides a flexible coding scheme which can adapted itself to different sources (audio or speech) and bitrates (no need to store quantization tables).

This chapter is organized as follows. In Section 5.1 the principles of stack-run coding are described and an example of stack-run mapping for an integer sequence is given. Then in Section 5.2 the proposed transform audio coder with stack-run coding is presented. The encoder and the decoder are described separately. Also we give the bit allocation in order to transmit parameters from the encoder to the decoder and the rate control for stack-run coding is explained. In Section 5.3, the proposed coder is modified by introducing a model-based deadzone into the scalar quantizer. Objective and subjective quality results are presented in Section 5.4 before concluding in Section 5.5.

5.1 Stack-Run Coding

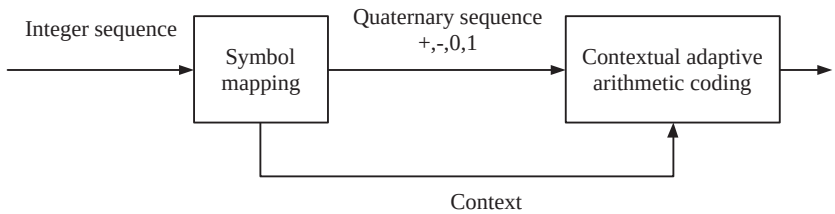


Figure 5.2: Principle of stack-run coding.

Stack-run coding [122, 121, 125, 95] is originally a lossless coding method applied to wavelet image coding. Fig. 5.2 presents the principle of stack-run coding. An integer sequence is mapped into a quaternary sequence (+, -, 0, 1) which is coded by a contextual adaptive arithmetic coding.

In this Section, we present the mapping rules in order to have a quaternay sequence from an integer sequence, following by the rules for the adaptive arithmetic coding to differenciate the contexts.

5.1.1 Representation of an integer sequence by a quaternary alphabet

The integer sequence is partionned into two contexts: sequence of zeros ("runs") and non-zero integers ("stack").

A stack is a non-zero integers written in binary format. So, we get a column of bits with the most significant bit (MSB) at the top and the least significant bit (LSB) at the bottom. This binary representation is unsigned and sign information is considered apart.

A run is a sequence of zeros, the length of the sequence is written in binary format which is ordered from LSB to MSB.

The symbols alphabet have the following meanings [121, 122]:

- "0" is used to signify a bit value of 0 in stack encoding.
- "1" is used for a bit value of 1 in stack encoding, but it is not used for the MSB (symbols "+" or "-" to give the stack sign).
- "+" is used to represent the positive MSB of stack and for a bit value of 1 in representing run lengths.
- "-" is used to represent the negative MSB of stack and for a bit value of 0 in representing run lengths.

Only "+" and "-" symbols are used to code runs, "0" and "1" are used to coded a bit value for a stack. The MSB of each stack is replaced by "+" if the associate coefficient is positive and "-" if it is negative. The mapping rules ensure that the stack-run conversion is reversible.

5.1.2 Context-based adaptive arithmetic coding

In order to distinguish a stack of absolute value 1, which will be represented only by "+" or "-" if we considered the mapping rules, from a run the absolute value of stack is incremented by one. For example the binary representation of +1 is "+0" instead of "+" and the binary representation of -8 is "-001" instead of "-000".

In representing stack, the symbols "+" and "-" are used simultaneously to encode the sign coefficients and bit plane location of the MSB. Another gain can be obtained in representation of the run lengths, which are ordered from LSB to MSB. Because all binary run lengths start with 1, the final "+" (MSB) can be omitted for most run-lengths representation without loss of information. The only potential problem is the representation of a run of length one which will not be representable if all MSB "+" symbols were eliminated. In order to avoid this problem, the MSB "+" symbols are retained for all runs of length $2^k - 1$ where k is a positive integer. For example a run length of value 10 is represented by "-+-" instead of "-+-+".

For the contextual arithmetic coding, the run context is the symbol for the run lengths associated with the LSB of the stack and the stack context is the rest of the

stack symbols. The coder is always switching from one context to the other one. In the special case where there is no run length between two stack values the run context is the LSB of the second stack.

The performance of the contextual arithmetic coding is increased by the fact that the probability tables used in the arithmetic coding can be adaptive. Also the run and the stack context can be considered separately which means having two probability tables that can be independently adapted.

5.1.3 Example of stack-run coding

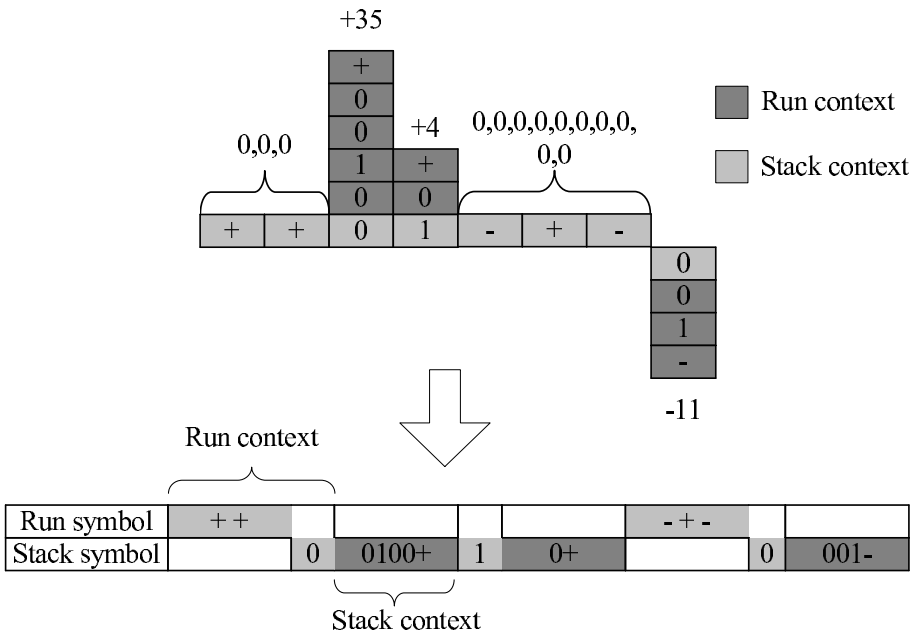


Figure 5.3: Example of stack-run coding.

Fig. 5.3 presents an example of mapping integer coefficients into quaternary symbol stream (+,-,0,1) and then how the symbols are mapped into contexts. The integer sequence taken as an example is $\{000, +35, +4, 0000000000, -11\}$.

The integer sequence starts with a run of 3 zeros, the binary representation of 3 is "11", which gives the run symbol "++". Because the length of the run is $2^k - 1$ the final "+" (MSB) is not dropped. The next coefficient is a stack of decimal value 35 which is incremented by one to 36. The binary representation of 36 is "001001" from LSB to MSB. So, the stack symbol of +35 is "00100+". Then, the next coefficient is again a stack of decimal value 4 which gives a stack symbol "10+". After we have a run of 10 zeros correspond to a binary representation "0101". The run symbol is then "-+-+", but in this case the final "+" is redundant (the run length is not $2^k - 1$ so the final "+" (MSB) can be dropped), so this run of 10 zeros is represented by "-+-". The final coefficient is a stack of value -11, the absolute value is incremented

by one to 12. The binary representation of 12 is "0011", because the number is negative the stack symbol is "001-".

The first context of this sequence is a run context equal to "++0" corresponding to the run symbol "++" associated with the LSB of the stack symbol after "0". Then there is a stack context "0100+" corresponding to the rest of the stack symbol. After there is no run between two stack so the run context is equal to "1" corresponding to the LSB of the next stack symbol.

5.2 Transform Audio Coding with Stack-Run Coding

In this section, we present a new transform audio coder with stack-run coding and model-based rate control.

5.2.1 Proposed encoder

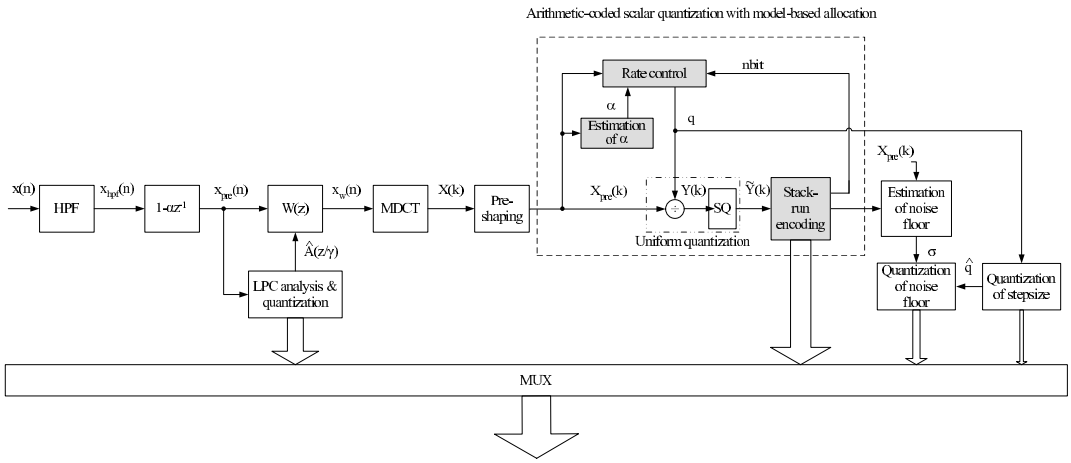


Figure 5.4: Block diagram of the proposed predictive transform encoder.

The proposed encoder is illustrated in Figure 5.4. The boxes in grey are the novelty compared to a standard transform coding. The encoder employs a linear-predictive weighting filter followed by MDCT coding. The input sampling frequency is 16000 Hz, while the frame length is 20 ms with a lookahead of 25 ms. The effective bandwidth of the input signal is considered to be 50-7000 Hz. An elliptic high-pass filter (HPF) is applied to the input signal $x(n)$ in order to remove the frequency component under 50 Hz. This pre-processing is necessary for the LPC analysis. The resulting signal $x_{hpf}(n)$ is then preemphasized by $1 - \alpha z^{-1}$ with $\alpha = 0.75$ to reduce the dynamic range of the input signal. An 18th order LPC analysis described in chapter 4 (see also [77]) is performed on the preemphasized signal $x_{pre}(n)$. The resulting LPC coefficients are quantized with 40 bits using a parametric method

based on a Gaussian mixture model (GMM) [113] in the linear spectrum frequency (LSF) domain. The pre-emphasized signal is filtered by a perceptual weighting filter:

$$W(z) = \frac{\hat{A}(z/\gamma)}{1 - \beta z^{-1}} \tag{5.1}$$

where $\beta = 0.75$ is a tilt parameter and $\gamma = 0.92$. The coefficients of $W(z)$ are updated every 5 ms by interpolating LSF parameters to smooth the evolution of the spectral envelope. A MDCT analysis is applied on the weighted signal $x_w(n)$. The MDCT is implemented using the fast algorithm of [30] which is based on a complex FFT. Then the MDCT coefficients $X(k)$ are pre-shaped to emphasize low frequencies, similar to 3GPP AMR-WB+ [7]. Preshaping [11] is used to correct the imperfect masking of $W(z)$, in particular for signals with a frequency around 300-500 Hz. The pre-shaped coefficients $X_{pre}(k)$ are divided by a step size q and the resulting spectrum $Y(k)$ is quantized by uniform scalar quantization with stepsize q. For a given spectrum $Y(k)$ the spectrum $\tilde{Y}(k)$ after quantization is defined as:

$$\tilde{Y}(k) = [Y(k)] = \left[\frac{X_{pre}(k)}{q}\right] \tag{5.2}$$

where [.] represents the rounding to the nearest integer. Only the first 280 coefficients of the $Y(k)$ spectrum corresponding to the 0-7000 Hz band are coded; the last 40 coefficients are discarded. The integer sequence $\tilde{Y}(k)$ is encoded by stack-run coding [122].

The rate control consists in finding the appropriate step size q so that the number of bits, *nbit*, used for stack-run coding matches the asymptotic allocated bit budget as described in Section 3.4. The distribution of $X_{pre}(k)$ is approximated by a generalized Gaussian model and the shape parameter α is estimated using Mallat's method presented in Section 3.2.2 (see also [69]). Then a noise level estimation is performed on the spectrum $Y(k)$ after stack-run coding. The noise floor σ is estimated as:

$$\sigma = r.m.s. \{X_{pre}(k) \,|\, Y(k) = 0\} \tag{5.3}$$

with the additional constraint that $Y(k)$ must belong to a long zero run to be really considered in the above r.m.s. calculation. The step size q is scalar quantized in log domain with 7 bits. The noise floor σ is quantized by coding the ratio $\sigma/\hat{q}$ in linear domain with 3 bits.

5.2.2 Bit allocation

Static distributions of bits to parameters

The parameters of the proposed coder are the Line Spectrum Frequency (LSF) parameters, the step size, and the noise floor level. The bit allocation to the parameters is detailed in Table 5.1, where B_{tot} is the total number of bits per frame. For instance at 24 kbit/s, $B_{tot} = 480$ bits. The allocation (in bits per sample) to stack-run coding is $B = (B_{tot} - 50)/280$, where 280 is the number of coefficients

in each frame and $50(40 + 7 + 3)$ is the number of bits needed to transmit the parameters to the decoder.

Table 5.1: Bit allocation for the stack-run predictive transform audio coding.

Parameter	Number of bits
LSF	40
Step size q	7
Noise floor σ	3
Stack-run coding	B_{tot}-50
Total	B_{tot}

Rate control for stack-run coding

Fig. 5.5 gives the bit allocation for the stack-run coding before the rate control. This example is done on a female French sample of 8 seconds coded at 24 kbit/s. As we can see, there is a biais between the number of bit using highrate estimated stepsize and the number of bits per frame. This biais of approximatively 84 bits could be explained by two types of mismatch:

- The highrate assumption made in the asymptotic model-based bit allocation is not verified. At 24 kbit/s the bit budget is 270 bits per frame which gives an average bitrate of $430/280 \approx 1.5$ bits/sample.
- The theory supposed that we have an ideal entropy coder which minimizes the entropy of order zero. But in practice we have an adaptive stack-run coding which minimizes entropy of higher order.

A bisection search for rate control is done in order to be within bit budget constraint for stack-run coding.

5.2.3 Proposed decoder

The decoder in error-free conditions is illustrated in Figure 5.6. The decoded LSF parameters are interpolated every 5 ms and converted to LPC coefficients. The reconstructed spectrum $\hat{Y}(k)$ is given by:

$$\hat{Y}(k) = \hat{q}\ \tilde{Y}(k) \tag{5.4}$$

where $\tilde{Y}(k)$ is found by stack-run decoding. In order to improve quality, in particular to avoid musical noise, a noise injection is applied on $\hat{Y}(k)$. A noise of magnitude $\pm\hat{\sigma}$ is injected in all zero sequences longer than 20 coefficients in $\hat{Y}(k)$. The spectrum $\hat{X}(k)$ is de-shaped in a way similar to 3GPP AMR-WB+ and transformed in time domain using the inverse MDCT and overlap-add algorithm described in [30]. An inverse perceptual filter $W(z)^{-1}$ is applied on $\hat{x}_w(n)$ in order to shape the coding noise introduced in the MDCT domain. The response of $W(z)^{-1}$ is similar to a short-term masking curve and its coefficients are updated every 5 ms by LSF interpolation. The signal $\tilde{x}(n)$ is deemphasized to find the synthesis $\hat{x}(n)$.

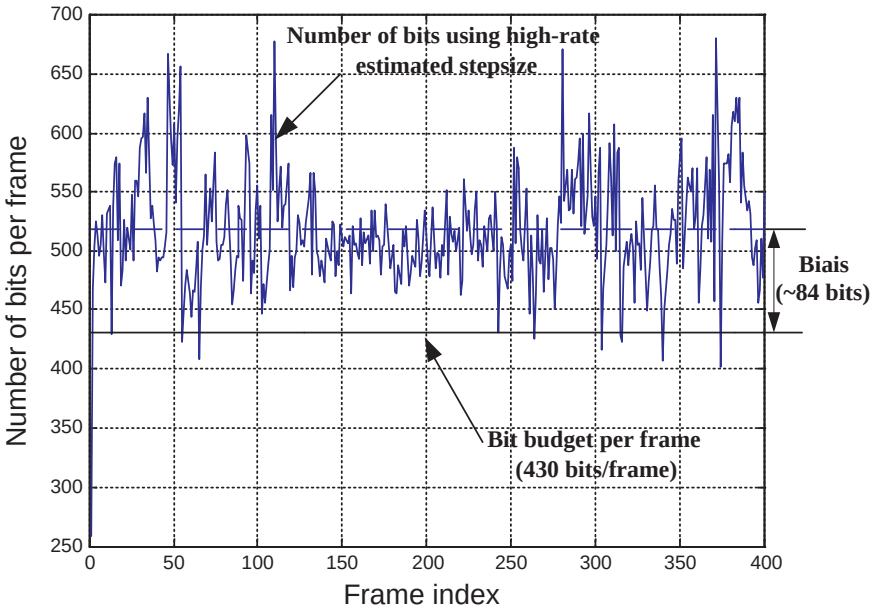


Figure 5.5: Bit allocation before the rate control (24 kbit/s) .

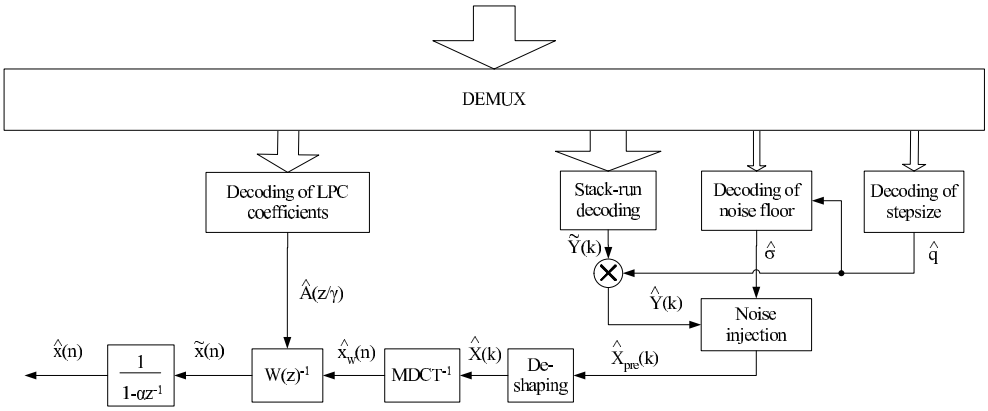


Figure 5.6: Block diagram of the proposed predictive transform decoder.

5.3 Inclusion of a Model-Based Deadzone

The encoder described in Section 5.2 is modified to include a uniform scalar deadzone quantizer and the size of the deadzone is based on the model. Fig. 5.7 presents the block diagram of the proposed predictive transform encoder with a deadzone quantizer. The deadzone optimization is based on the pdf of $X_{pre}(k)$ as described in Section 3.1.3 (see also [88]). The pre-shaped spectrum $X_{pre}(k)$ is divided by stepsize q and the resulting coefficients $Y(k)$ are encoded by scalar quantization

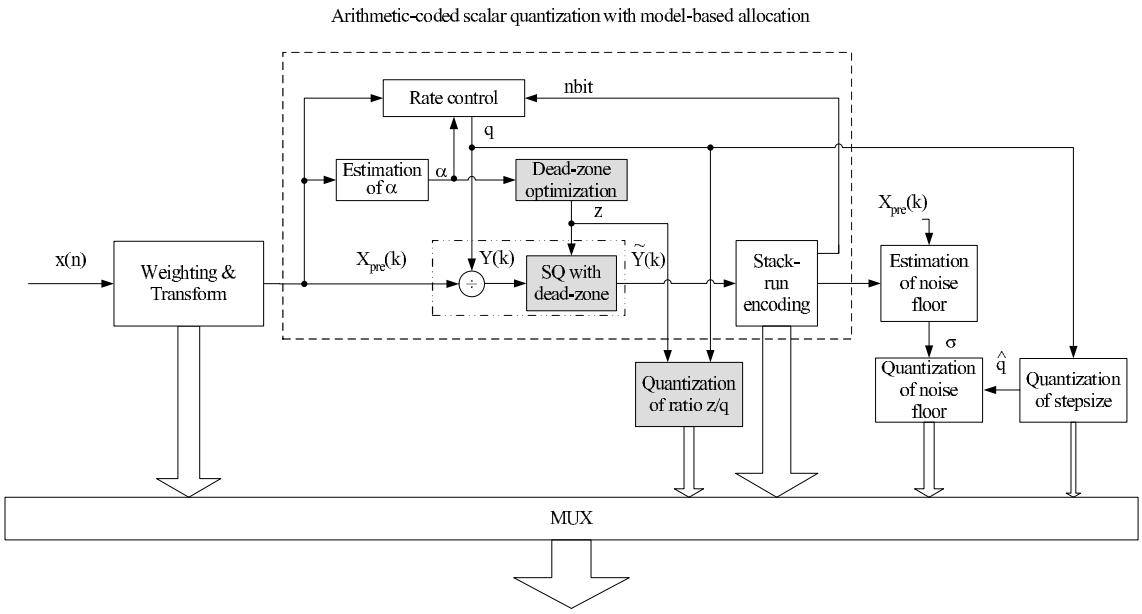


Figure 5.7: Block diagram of the proposed predictive transform encoder with a deadzone quantizer.

with deadzone:

$$\tilde{Y}(k) = \begin{cases} X_{pre}(k)/q - (z-q)/2q & \text{if } X_{pre}(k) > z/2 \\ X_{pre}(k)/q + (z-q)/2q & \text{if } X_{pre}(k) < -z/2 \\ 0 & \text{otherwise} \end{cases} \tag{5.5}$$

The inclusion of a deadzone for quantization was first introduced in [114] for Laplacian distribution. It has shown that under high rate assumption the optimal deadzone z is close to the stepsize q. In the case of low bitrate for a Laplacian distribution the optimal deadzone z is two times the stepsize q [70].

In order to transmit the deadzone z to the decoder, the ratio between the deadzone z and the stepsize q is scalar quantized with 2 bits. Table 5.2 presents the modified bit allocation for stack-run coding with $z = q$, $z = 2q$, or $z = z_{opt}$ for a quantizer with a centroid set to mid-value.

If the centroid of the quantizer is not set to mid-value, we need 2 more bits in order to transmit it. So, the bit allocation presented in Table 5.2 is modified and the number of bits used to transmit the parameters is 52 or 54 depending on the case.

5.3.1 Optimization of the deadzone

We consider the encoding of N zero-mean generalized Gaussian variables $x_1, \ldots, x_N$ of variances σ^2 with respect to the mean square error criterion. We assume that the variables x_i have a generalized Gaussian pdf $g_{\sigma,\alpha}(x)$ of shape parameter α. The

Table 5.2: Bit allocation for the coding scheme with deadzone.

Parameter	Number of bits
LSF	40
Step size q	7
Noise floor σ	3
Stack-run coding with $z = q$	B_{tot}-50
Stack-run coding with $z = 2q$	B_{tot}-50
Ratio z/q	2
Stack-run coding with $z = z_{opt}$	B_{tot}-52
Total	B_{tot}

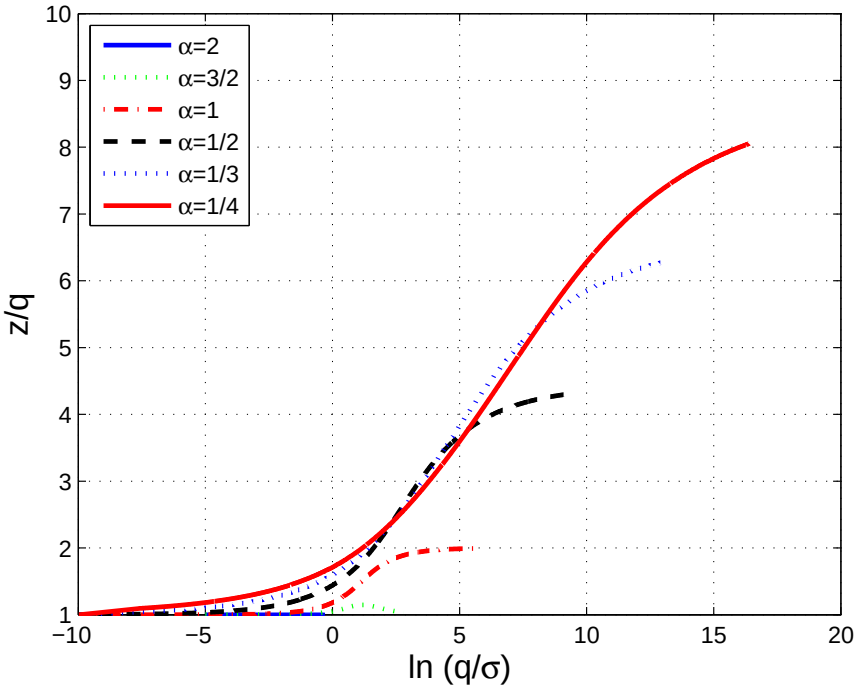


Figure 5.8: Optimal deadzone for an uniform scalar quantizer with centroid (assuming a generalized Gaussian model).

variables x_i are coded by deadzone scalar quantization with the *same* step size q and so the *same* dead-zone length z. We assume that the sequence of integers obtained after deadzone scalar quantization is encoded by ideal entropy coding.

The distortion is given by [88]:

$$D\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right) = 1 - 2\sum_{m=1}^{+\infty} \frac{f_{1,m}\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right)}{f_{0,m}\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right)} \tag{5.6}$$

where α is the shape parameter of x_i and $f_{n,m}\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right)$ is a function defined as:

$$f_{n,m}\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right) = \int_{z/2\sigma+(m-1)q/2\sigma}^{z/2\sigma+mq/2\sigma} x^n g_{1,\alpha}(x)dx \tag{5.7}$$

The bit rate is given by [88]:

$$\begin{aligned} b\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right) &= -f_{0,0}\left(\alpha, \frac{z}{\sigma}\right) \log_2 f_{0,0}\left(\alpha, \frac{z}{\sigma}\right) \\ &\quad -2\sum_{m=1}^{+\infty} f_{0,m}\left(\alpha, \frac{z}{\sigma_i}, \frac{q}{\sigma}\right) \log_2 f_{0,m}\left(\alpha_i, \frac{z}{\sigma}, \frac{q}{\sigma}\right) \end{aligned} \tag{5.8}$$

where $f_{0,0}\left(\alpha, \frac{z}{\sigma}\right)$ is a function defined as:

$$f_{0,0}\left(\alpha, \frac{z}{\sigma}\right) = \int_{z/2\sigma}^{z/2\sigma} x^n g_{1,\alpha}(x)dx \tag{5.9}$$

For a given bit allocation B in bits per sample, the bit allocation problem is to minimize the distortion D under the constraint that $\sum_{i=1}^{N} b\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right) \leq B$. The distortion $D\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma_i}\right)$ can be minimized with Lagrangian techniques. The criterion $J(z, q, \lambda)$ is defined as

$$J(z, q, \lambda) = \sum_{i=1}^{N} D\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right) - \lambda\left(\sum_{i=1}^{N} b\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right) - B\right) \tag{5.10}$$

where λ is the Lagrange multiplier. It can be shown that the optimal deadzone z is given by the solution to the equation [89]:

$$\boxed{\frac{\frac{\partial D}{\partial z}\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right)}{\frac{\partial b}{\partial z}\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right)} = \frac{\frac{\partial D}{\partial q}\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right)}{\frac{\partial b}{\partial q}\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right)}} \tag{5.11}$$

From Eq. 5.11 we have a relationship between $ln\,(q/\sigma)$ and z/q. So for practical implementation, we store tables of this relationship for shape parameters α. Fig. 5.8 presents charts in order to have the optimal deadzone z depending of the shape parameter α, the stepsize q and the variance σ. As we can see, when q is getting smaller (high bitrate) we have $z = q$. For lower bitrate, z is more important than q which means that we have more zeros.

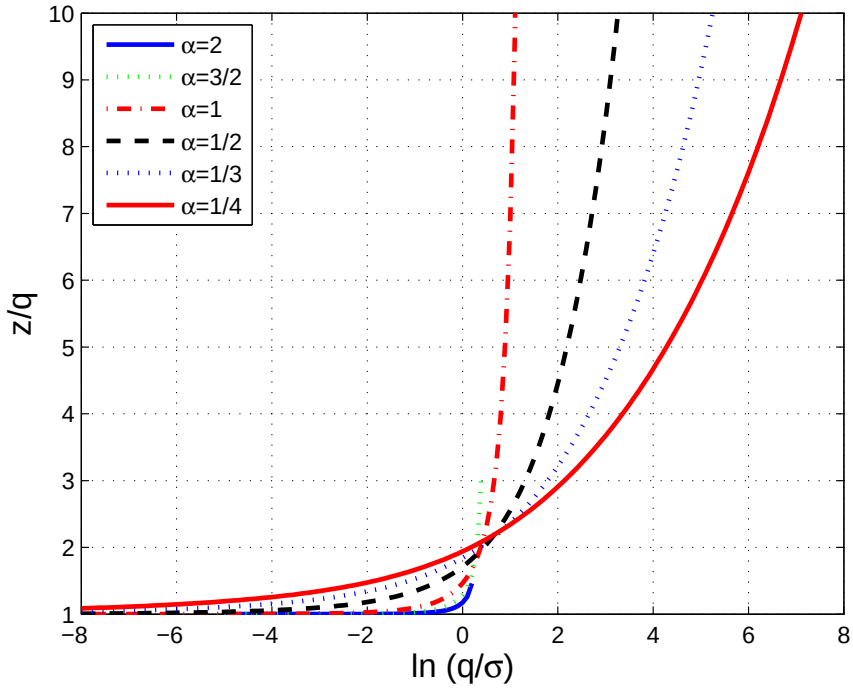


Figure 5.9: Optimal deadzone for an uniform scalar quantizer with centroid set to mid-value (assuming a generalized Gaussian model).

5.3.2 Optimization of the deadzone for a centroid set to mid-value

If the reconstruction level of the quantizer is set to mid-value the distortion D is given by [88]:

$$\begin{aligned} D\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right) &= 1 + 2\sum_{m=1}^{+\infty}\left(\frac{z}{2\sigma} + \left(m - \frac{1}{2}\right)\frac{q}{\sigma}\right)^2 f_{0,m}\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma}\right) \\ &\quad -4\sum_{m=1}^{+\infty}\left(\frac{z}{2\sigma} + \left(m - \frac{1}{2}\right)\frac{q}{\sigma}\right) f_{1,m}\left(\alpha, \frac{z}{\sigma}, \frac{q}{\sigma_i}\right) \end{aligned} \tag{5.12}$$

Fig. 5.9 presents charts in order to have the optimal deadzone z depending of the shape parameter α, the stepsize q and the variance σ.

5.4 Experimental Results and Discussion

A database of 24 clean speech samples in French language (6 male and female speakers×4 sentence-pairs) and 16 clean music samples (4 types×4 samples) of 8

seconds is used. These samples are sampled at 16 kHz, preprocessed by the P.341 filter of ITU-T G.191A and normalized to -26 dB_{ov} using the P.56 speech voltmeter.

5.4.1 Optimization of the deadzone

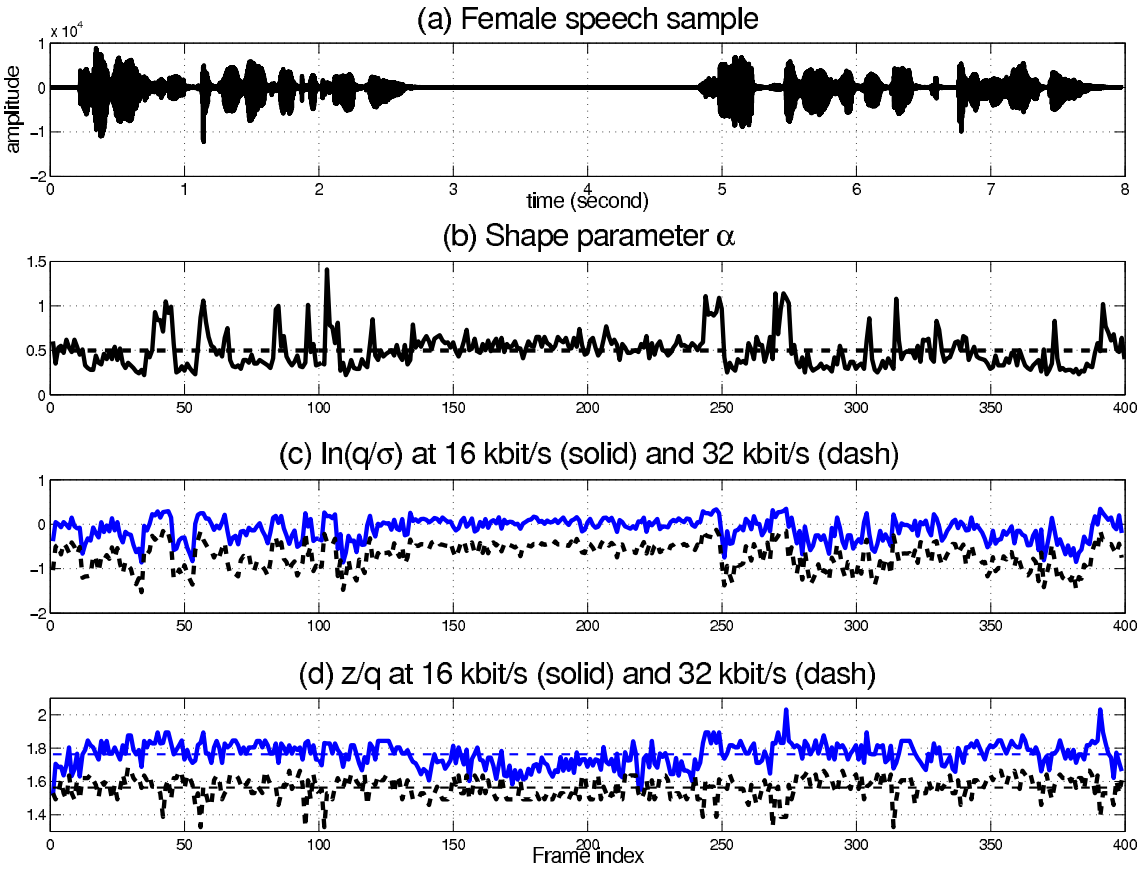


Figure 5.10: Example of dead-zone optimization on a French female speaker sample.

We presented in Fig. 5.10 an example of the dead-zone optimization with a centroid at middle-value for a French female speaker sample of 8 seconds at two bitrates 16 and 32 kbit/s. As we can see in Fig. 5.10 (c) $ln(q/\sigma)$ at 32 kbit/s is smaller than the one at 16 kbit/s because the stepsize at high bitrate is smaller than the one at low bitrate. Also in Fig. 5.10 (d) the mean value of $z/q \approx 1.6$ at 32 kbit/s and the mean value of $z/q \approx 1.8$ at 24 kbit/s which confirm the theory that at high bitrate z/q is getting closer to 1.

5.4.2 Spectral distortion statistics for LPC quantization

The performance of the LSF quantization is evaluated with the spectral distortion [86] defined as :

$$SD = \sqrt{\frac{1}{2\pi}\int_{-\pi}^{\pi}\left[10\log_{10}\frac{1}{|A(\frac{e^{j\omega}}{\gamma})|^2} - 10\log_{10}\frac{1}{|\hat{A}(\frac{e^{j\omega}}{\gamma})|^2}\right]^2 d\omega} \tag{5.13}$$

where $1/|A(e^{j\omega}/\gamma)|^2$ and $1/|\hat{A}(e^{j\omega}/\gamma)|^2$ are respectively the original LPC spectrum and the quantized LPC spectrum. The spectral distortion (SD) for the quantization of $A(z/\gamma)$ with 40 bits is presented in Table 5.3. The average SD is around 1.20 dB and the amount of outliers is limited. This LPC quantization is not exactly transparent but the related spectral distortion is quite acceptable.

Table 5.3: Spectral distortion for the quantization of $A(z/\gamma)$ with 40 bits.

avg. SD (dB)	$SD \geq 2$ dB (%)	$SD \geq 4$ dB (%)
1.20	15.17	1.04

5.4.3 Objective quality results

WB-PESQ [84] is used to evaluate the quality of the proposed coder and compare it with ITU-T G.722.1. Only clean speech samples are used to compute the average WB-PESQ scores at various bitrates. The bit rate varies from 16 to 40 kbit/s with a step of 4 kbit/s for our coder. ITU-T G.722.1 is tested at 24 and 32 kbit/s. Fig. 5.11 and 5.12 show the WB-PESQ scores obtained for the two coders, by considering separately male and female cases. These results suggest that the quality of the proposed coder is better than ITU-T G.722.1 at 24 and 32 kbit/s (0.2-0.3 MOS-LQ0 difference).

As we can see in Fig. 5.11, using a scalar quantizer with $z = z_{opt}$ or a $z = 2q$ improves the performance at low bitrate. It seems that having $z = z_{opt}$ or $z = 2q$ is equivalent, it could be explain by the fact that we need two bits to transmit z_{opt} which is not the case for $z = 2q$ or $z = q$.

Fig. 5.12 confirms the results of Fig. 5.11. We also see that having a centroid or a centroid set to middle value does not change the WB-PESQ score significantly. Using a centroid give more precision but it also costs two more bits in order to transmit it to the decoder.

5.4.4 Subjective quality results

Comparison of stack-run coding and ITU-T G.722.1

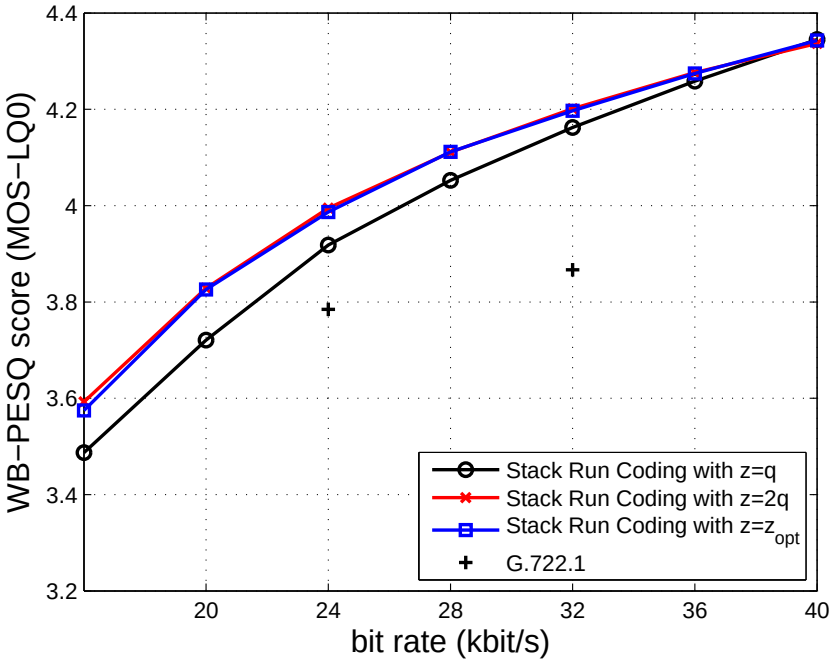


Figure 5.11: Average WB-PESQ score (centroid set to mid-value)with noise injection.

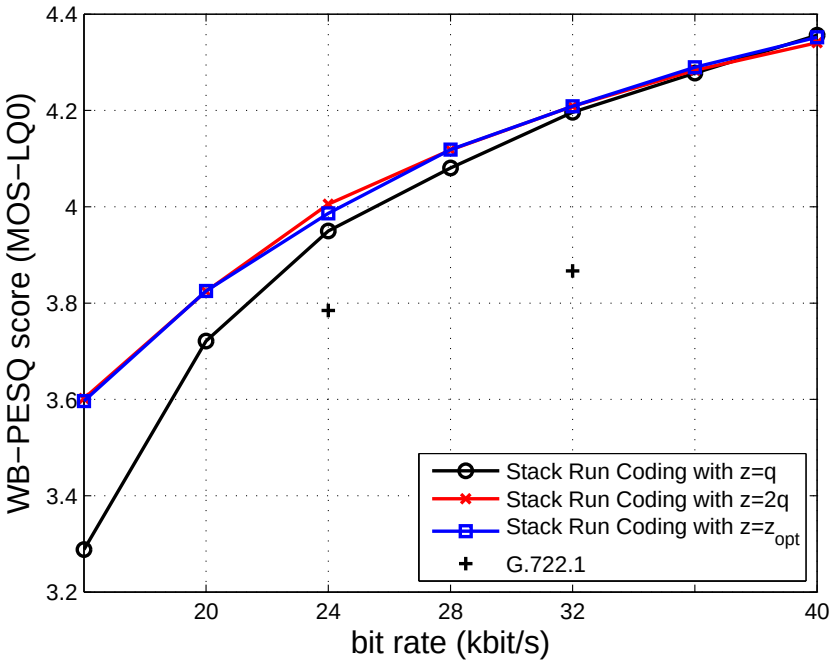


Figure 5.12: Average WB-PESQ score (centroid) with noise injection.

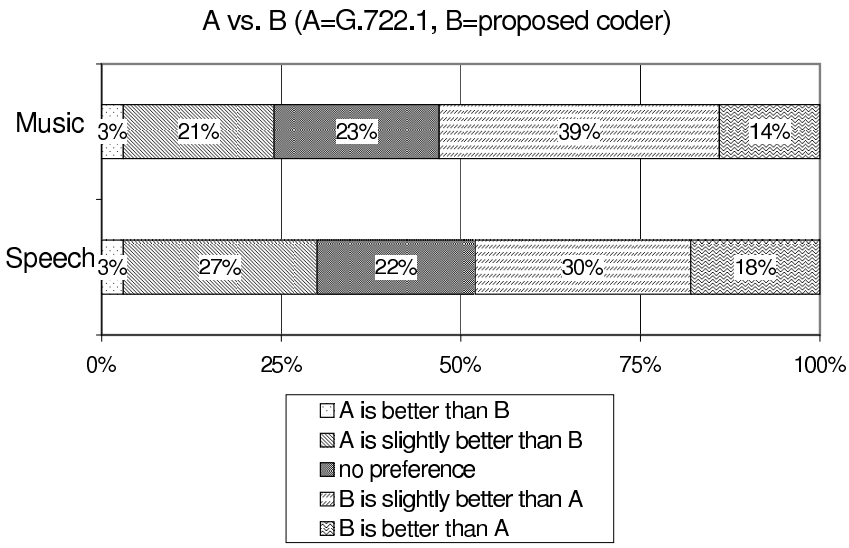


Figure 5.13: AB test results at 24 kbit/s.

Two informal AB tests at 24 kbit/s have been conducted: one for speech, another for music. In total 8 experts participated in the test. Fig. 5.14 shows the results. The proposed coder was preferred for music in 53% of cases and for speech in 48% of cases. The results confirmed the objective quality results at 24 kbit/s. Informal subjective tests have also been conducted at 32 kbit/s but the quality improvement of the proposed coder is less significant and the two coders are equivalent. The proposed coder is better than G.722.1 at 24 kbit/s and equivalent at 32 kbit/s, but it has a higher complexity than G.722.1.

The discrepancy between objective and subjective results at 24 and 32 kbit/s can be explained by the large sensitivity of WB-PESQ in low frequencies – indeed the proposed coder has in general a lower distortion in low frequencies –, and by the limited applicability of WB-PESQ to compare distortions of different natures.

Comparison of stack-run coding with $z = q$ and $z = z_{opt}$

An informal AB tests at 24 kbit/s has been conducted for speech in order to compare the stack-run coding with or without deadzone . In total 9 experts participated in the test. Fig. 5.14 shows the results. Stack-run coding with $z = z_{opt}$ (centroid set to mid-value) was preferred for speech in 50% of cases. The results confirmed the objective quality results at 24 kbit/s. Informal subjective tests have also been conducted at 32 kbit/s and the two coders are equivalent. Consequently, the use of an optimized deadzone z_{opt} does improve slightly quality, especially at low bitrates.

5.4.5 Delay and complexity

This model-based transform audio coder has been implemented in language C. The algorithmic complexity of stack-run coding is 45 ms (20 ms for the frame, 20ms for

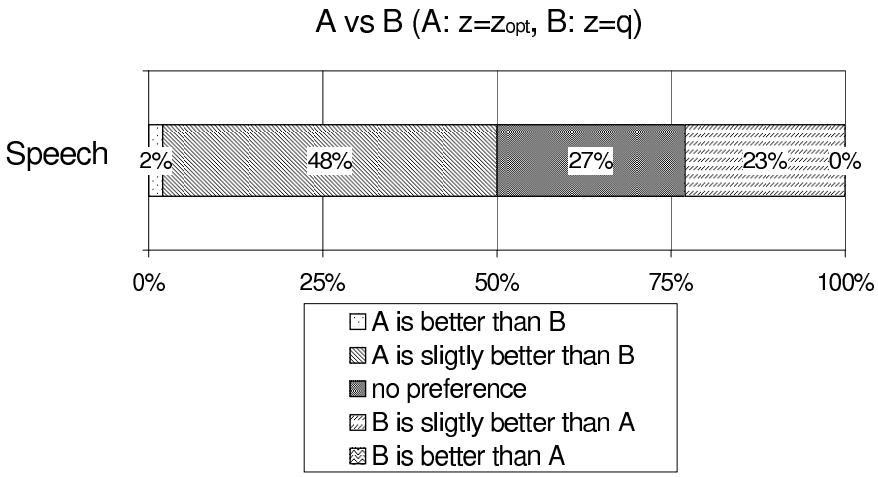


Figure 5.14: AB test results for speech at 24 kbit/s.

the MDCT and 5 ms for the lookahead), while that of G.722.1 is 40 ms.
The computational complexity of G.722.1 is low which is not the case with the stack-run coding. Indeed in the latter case, a rate control loop is needed [79] and in practice stack-run coding is performed several times per frame.
The memory requirements (in terms of data ROM) for the stack-run coding consists mainly of the storage of GMM parameters for LPC quantization and MDCT computation tables. To compute the optimal deadzone z_{opt} for each MDCT frame we have to store the tables which gives the relationship between $\ln(q/\sigma)$ and z/q.

5.5 Conclusion

In this chapter we proposed an MDCT coder with generalized Gaussian modeling for wideband speech and audio signals sampled at 16 kHz. This coder was compared with ITU-T G.722.1. The quality improvement is mainly due to the use of arithmetic coding (instead of Huffman coding) and perceptual filtering. The generalized Gaussian model allows to minimize the complexity of bit allocation by estimating efficiently the quantization stepsize.

We also proposed a non-asymptotic method to have optimal deadzone for scalar quantization, assuming that the distribution of MDCT coefficients is approximated by a generalized Gaussian model. In fact, stack-run coding with $z = 2q$ is near-equivalent to $z = z_{opt}$. This result relies on the assumption of generalized Gaussian modeling and the use of ideal entropy coding. Still, we can consider that $z = 2q$ is a general solution for scalar deadzone optimization and it is not specific to stack-run coding. Finally this result confirms [114]. However we do not have a Laplacian distribution and we do not assume high bitrate.

Chapter 6

Flexible Embedded Transform Coding based on Bit Plane Coding

Nowadays many speech and audio coding standards are available. Often they are optimized for specific constraints and used in specific applications. Besides, multimedia communication has to deal with the problem of heterogeneity of access (e.g. mobile, WiFi, DSL, FTTH) and terminals (narrowband/wideband, smartphone/softphone).

Based on this context, this work aims at reaching more flexibility in speech and audio coding and high coding efficiency. Specifically, we propose here an embedded coding method similar to the bit plane coding used for instance in MPEG-4 BSAC [90, 31, 64] for audio and JPEG2000 [119] for images. Furthermore, we use a model-based approach. Model-based coding has already shown promising results in speech and audio coding for LSF parameters [113], waveform coding of speech [105], coding of transform coefficients [79] and entropy-constrained vector quantization [129].

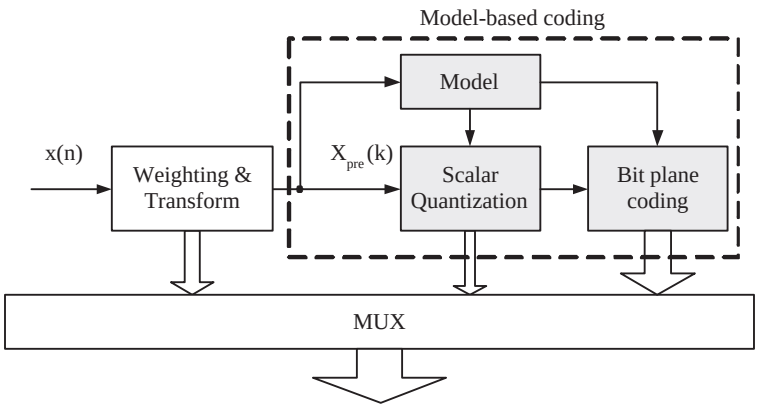


Figure 6.1: Proposed model-based transform coder.

Fig.6.1 presents the principle of the proposed model-based transform coding. The box in grey is the novelty compared to a standard transform coding. An input signal $x(n)$ is mapped into a "perceptual" domain by weighting and transform operations described in Section 6.4. The distribution of pre-shaped coefficients $X_{pre}(k)$

is approximated by generalized Gaussian modeling as described in Section 3.2. The information from the model are then used for the quantization and the arithmetic coding. A scalar quantization based on the model is applied on the coefficients $X_{pre}(k)$ and finally they are coded by model-based arithmetic coding, here we are using bit plane coding. The main difference with the coder presented in Chapter 5 is that the model is also used for arithmetic coding.

The main contributions of this work lies in the use of bit plane coded scalar quantization and the application of generalized Gaussian model for probabilities initialization of the bit plane coding.

This chapter is organized as follows. A review of related work to bit plane coding is first presented in Section 6.1. Bit plane coding is presented in Section 6.4. Then in Section 6.3 the model-based initialization of probabilities for adaptive arithmetic coding is presented. Three methods are presented, two conditional probabilities estimation and one non-conditional. Conditional probability used the information from bit planes previously encoded. In Section 6.4, the proposed embedded transform coding based on bit plane coding is explained. Objective and Subjective results are presented in Section 6.5 before concluding in Section 6.6.

6.1 Related work

In image coding, zero-tree quantization described first in [109] and later refined with the set partitioning in hierarchical trees (SPIHT) algorithm [101], combines ordered bitplane coding with significance maps [109]. This algorithm is based on the idea that spectral component with more energy content should be transmitted before other components. JBIG [115] is a lossless image compression of one-bit-per-pixel image data using an arithmetic coding knows as the Q-coder [91]. It bases the probabilities of each bit on the previous bits and the previous lines of the picture. In JPEG2000 [118, 120, 119], an embedded block coding algorithm based on discrete wavelet transform (DWT) subbands with dead-zone quantization and bit plane coding is used.

In audio coding, ordered bit plane coding is used in MPEG-4 BSAC [90, 59] as described in Section 2.4.2. But, BSAC coder requires the use of arithmetic coding which increase computational complexity, and the bitrate granularity is limited to 1 kbit/s. Also, an Embedded Audio Coding (EAC) [64] based on a embedded bit plane coding with an implicit auditory masking is used for audio storage and streaming. It performs better than the audio standard such as MP3 [1, 2]. An algorithm based on the SPIHT zero tree coding with a new method of arranging transform coefficients [31] is used for audio coding.

In MPEG-4 video standard [65], the quantized discrete transform (DCT) coefficients are coded using a bit plane coding method.

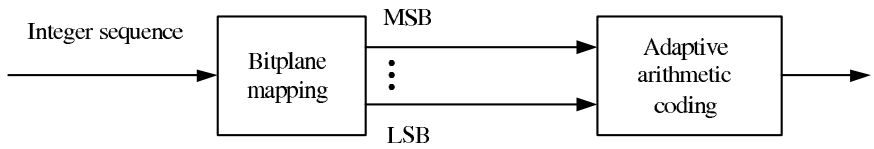


Figure 6.2: Principle of bit plane coding.

6.2 Bit Plane Coding

Bit plane coding of transform coefficients is an alternative approach to have scalability. In each frame, bit plane coefficients are coded in order of significance, beginning with the MSB and progressing to the LSB. This gives the possibility of a fine-grain scalability. Also it gives the possibility of encoding and decoding to different bitrates, so there is no need to have a recursive bit allocation search in fixed rate.

Fig. 6.2 presents the principle of bit plane coding. An integer sequence is mapped into bit planes which are coding using an adaptive arithmetic coding [126, 62].

In this section we present the mapping rules in order to decompose an integer sequence into bit planes following by an example. Then we present different ways of formatting the bitstream in order to transmit bit planes to the decoder.

6.2.1 Bit plane mapping

We consider the encoding of N variables $X = [x_1, \ldots, x_N]$ of variances σ^2 with respect to the mean square error criterion. After uniform scalar quantization of X with stepsize q, we have an integer sequence $\tilde{Y} = [\tilde{y}_1, \ldots, \tilde{y}_N]$ defines as:

$$\tilde{Y}(k) = \left[\frac{X_{pre}(k)}{q}\right] \tag{6.1}$$

where [.] represents the rounding to the nearest integer. For bit plane coding, the integer sequence is written in binary format. First, the sign and the absolute value are separated as:

$$\tilde{y}_i = |\tilde{y}_i|(-1)^{s_i} \tag{6.2}$$

where $|.|$ is the absolute value and s_i is the sign bit of y_i defined as:

$$s_i = \begin{cases} 1 & \text{si } \tilde{y}_i \leq 0 \\ 0 & \text{si } \tilde{y}_i \geq 0 \end{cases} \tag{6.3}$$

Then, each absolute value $|\tilde{y}_i|$ is written in binary format given by:

$$|\tilde{y}_i| = \mathbf{B}_{K-1}(|\tilde{y}_i|)2^{K-1} + \ldots + \mathbf{B}_1(|\tilde{y}_i|)2^1 + \mathbf{B}_0(|\tilde{y}_i|)2^0 \tag{6.4}$$

where $\mathbf{B}_k(|\tilde{y}_i|)$ is the kth bit of the binary representation of $\tilde{y}_i$. K is the number of bit plane needed to represent the integer sequence $\tilde{Y}$ defined as:

$$K = \max(\lceil log_2(\max_{i=1,\ldots,n} |\tilde{y}_i|)\rceil, 1) \tag{6.5}$$

where $\lceil \cdot \rceil$ is the upper integer and $\log_2(0) = -\infty$.

The sign bit of zero is indefinite, so we choose $s_i = 0$ if $y_i = 0$. Considering the binary representation of y_i , the kth bit plane is given by:

$$\mathbf{P}_k = [\mathbf{B}_k(|\tilde{y}_0|)\ \mathbf{B}_k(|\tilde{y}_1|) \ldots \mathbf{B}_k(|\tilde{y}_{N-1}|)] \tag{6.6}$$

with $k = 0, \ldots, K-1$ and the bit plane of signs $\mathbf{S}$ is given by:

$$\mathbf{S} = [s_0\ s_1 \ldots s_{N-1}] \tag{6.7}$$

The upper bit plane $\mathbf{P}_{K-1}$ is the most significant bits (MSB) plane. The lower bit plane $\mathbf{P}_0$ is the least significant bits (LSB) plane.

The sign bit s_i is transmitted only if $\tilde{y}_i$ is different from zero. In order to have partial decoding, the sign bit s_i is transmitted as soon as one of decoded bit $\{\mathbf{B}_k(|\tilde{y}_i|)\}_{k=0,\ldots,K-1}$ is equal to one. The symbol $\{\mathbf{B}_k(|\tilde{y}_i|)\}_{k=0,\ldots,K-1}$ is said to be significant if its value is different from zero.

6.2.2 Example of bit plane mapping

Y=		-2	+5	-3	-6	+7	+3	0	+1	
		1	0	1	1	0	0	0	0	Bit plane of signs
K=3	2^2	0	1	0	1	1	0	0	0	P_2=MSB
	2^1	1	0	1	1	1	1	0	0	
	2^0	0	1	1	0	1	1	0	1	P_0=LSB

Figure 6.3: Example of bit plane mapping.

Fig. 6.3 presents an example of bit plane mapping of an integer sequence $\tilde{Y} = [-2, +5, -3, -6, +7, +3, 0, +1]$ with $N = 8$ variables. In this case the number of bit plane is $K = 3$. The MSB plane is defined as $\mathbf{P}_2 = [0, 1, 0, 1, 1, 0, 0, 0]$, the LSB plane is $\mathbf{P}_0 = [0, 1, 1, 0, 1, 1, 0, 1]$, and the bit plane of sign is $\mathbf{S} = [1, 0, 1, 1, 0, 0, 0, 0]$.

6.2.3 Different ways to format the bitstream

Coding is done on successive bit planes $\mathbf{P}_k$, it is possible to divide $\mathbf{P}_k$ into sub-planes in order to have a progressive decoding. Fig. 6.4 presents different ways to format the bitstream in order to transmit bit planes.

Bit planes following by sign bits

The bitstream format present in Fig. 6.4 (a) is the simplest one. The number of bit planes is transmitted first, following by codes (variable length) associated to

Nb of bitplane	P_0	P_1		P_{K-1}	S

(a) Bit planes following by sign bits.

Nb of bitplane	P_0	S_0	P_1	S_1		P_{K-1}	S_{K-1}

(b) Bit plane with "significant" signs bits.

Nb of bitplane	P_{00}	S_{00}	P_{01}	S_{01}		P_{0L-1}	S_{0L-1}	

(c) Sub-planes of the bit plane with "significant" sign bits.

Figure 6.4: Different bitstream format.

each bit plane $\mathbf{P}_k$. At the end, a code corresponding to bit plane of signs $\mathbf{S}$ is transmitted. This bitstream format is very easy to use but it does not give an efficient progressive decoding. If at the decoding we want to have a bitrate different from the encoding, the bit plane of signs $\mathbf{S}$ will be the first to be lost and so the decoded signal will not be intelligible. Also a code is associated to a specific bitplane $\mathbf{P}_k$. So if one code is deteriorated, $\mathbf{P}_k$ might not be intelligible and so the decoded signal.

Bit plane with "significant" signs bits

In order to do progressive decoding, we can use a bitstream format as proposes in Fig. 6.4 (b). Sign bits associated to a significant coefficient in $\mathbf{P}_k$ are transmitted immediately after $\mathbf{P}_k$, so we divide the bit plane of signs $\mathbf{S}$. In order to do it, we defined the sign bit of the ith coefficient in $\mathbf{P}_k$ as:

$$s_{k,i} = \begin{cases} 1 \text{ if } \mathbf{B}_k(|\tilde{y}_i|) = 1 \text{ and } \mathbf{B}_j(|\tilde{y}_i|) = 0 \ \forall j > K \\ \emptyset \text{ otherwise} \end{cases} \quad \forall k < K \text{ and } \forall i < N \tag{6.8}$$

The bit plane of signs $\mathbf{S}_k$ associated to $\mathbf{P}_k$, with $0 \leq k < K$, is defined as:

$$\mathbf{S}_k = [s_{k,0} \ s_{k,1} \ldots s_{k,i} \ldots s_{k,N-1}] \ \forall i < N \tag{6.9}$$

The only difference between Fig. 6.4 (a) and (b) is the bit plane of signs which is transmitted between codes of bit plane $\mathbf{P}_k$. So, instead of transmitting one code for $\mathbf{S}$, we transmit sub-codes associated to each $\mathbf{S}_k$. With this bitstream format, if the decoding bitrate is different than the encoding, we are not going to loose all the sign bits. But, to transmit the bit plane $\mathbf{P}_k$ we still have one code for a bit plane $\mathbf{P}_k$. So if the code is deteriorated, $\mathbf{P}_k$ might not be intelligible and so the decoded signal.

Sub-planes of the bit plane with "significant" sign bits

In order to avoid this problem of intelligibility, we propose to use a bitstream format as showed in Fig. 6.4 (c). The bit plane $\mathbf{P}_k$ is divided into sub-planes $\mathbf{P}_{k,j}$ of size L, where L is a submultiple of N:

$$\mathbf{P}_k = [\mathbf{P}_{k,0} \ \mathbf{P}_{k,1} \ldots \mathbf{P}_{k,j} \ldots \mathbf{P}_{k,J}] \tag{6.10}$$

where $J = N/L$. The sub-plane $\mathbf{P}_{k,j}$,$\forall i < L, \forall k < K$ and $\forall j < J$ is defined as:

$$\mathbf{P}_{k,j} = [\mathbf{B}_k(|\tilde{y}_{j\times L}|)\mathbf{B}_k(|\tilde{y}_{j\times L+1}|)\ldots\mathbf{B}_k(|\tilde{y}_{j\times L+i}|)\ldots\mathbf{B}_k(|\tilde{y}_{j\times L+L-1}|)] \tag{6.11}$$

The bit plane of signs $\mathbf{S}_k$ is divided like the bit plane $\mathbf{P}_k$ into sub-plane $\mathbf{S}_{k,j}$ of size L:

$$\mathbf{S}_k = [\mathbf{S}_{k,0}\mathbf{S}_{k,1}\ldots\mathbf{S}_{k,j}\mathbf{S}_{k,J}] \tag{6.12}$$

where J is the number of sub-plane $\mathbf{S}_{k,j}$ such that $J = N/L$. The sub-plane $\mathbf{S}_{k,j}$, $\forall i < L, \forall k < K$ and $\forall j < J$ is defined as:

$$\mathbf{S}_{k,j} = [s_{k,j\times L}s_{k,j\times L+1}\ldots s_{k,j\times L+i}\ldots s_{k,j\times L+L-1}] \tag{6.13}$$

The bit plane $\mathbf{P}_k$ and the bit plane of signs $\mathbf{S}_k$ are divided into sub-planes $\mathbf{P}_{k,j}$ and $\mathbf{S}_{k,j}$. So, instead of transmitting one cone for a given bit plane $\mathbf{P}_k$, we transmit sub-codes for each sub-planes $\mathbf{P}_{k,j}$. So if one of the sub-codes is deteriorated, we do not loose all information on the bit plane $\mathbf{P}_k$. It is possible to divide $\mathbf{P}_k$ in sub-plane until we have a sub-plane $\mathbf{P}_{k,j}$ of only one coefficient.

6.3 Proposed Bit-Sliced Arithmetic Coding with Model-Based Probabilities

The probability tables used for arithmetic coding are initialized with $p(0) = p(1)$ and they adapt themselves to the signal as we are encoding it. From a practical point of view, we only have 280 coefficients in each bit plane, so the adaptive arithmetic coding does not have enough time to adapt itself to the source. In order to solve this problem, we propose to approach the source with a generalized Gaussian model and to estimate the initial probability tables based on this model.

Firstly, we present the estimation of model-based probabilities on a generalized Gaussian distribution. Then we present 3 methods in order to initialize the probability tables for the bit plane coding. Two methods are estimating conditional probabilities and one is not conditional.

6.3.1 Estimation of model-based probabilities

Estimation of binary probabilities for bit plane is linked to the probability of y_i. We consider the encoding of N zero-mean random variables $X = [x_1, \ldots, x_N]$ of variances σ^2. After uniform scalar quantization of stepsize q, we have an integer sequence $\tilde{Y} = [\tilde{y}_1, \ldots, \tilde{y}_N]$. We assume that the variables x_i have a generalized Gaussian pdf $g_{\sigma,\alpha}(x)$ of shape parameter α. Then, the probability of having the coefficient $\tilde{y}_i$ is:

$$\boxed{p(\tilde{y}_i) = \int_{q\tilde{y}_i - q/2}^{q\tilde{y}_i + q/2} g_{\sigma,\alpha}(x)dx} \tag{6.14}$$

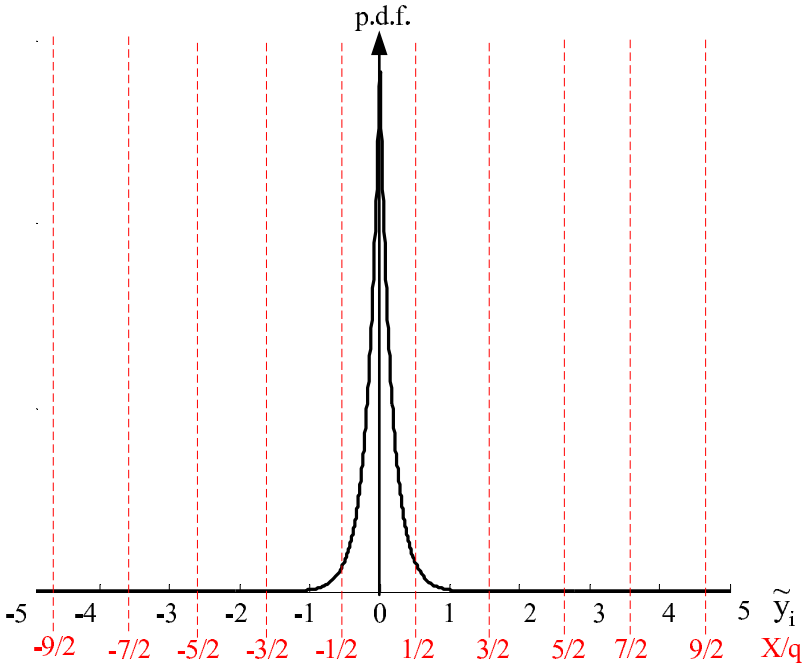


Figure 6.5: Example of probabilities estimation for a Laplacian distribution (zero mean unit variance) with uniform scalar quantization of stepsize q.

The generalized Gaussian distribution is symmetrical, so $p(\tilde{y}_i) = p(-\tilde{y}_i)$. We restrict ourselves to the case of a uniform scalar quantizer.

Fig. 6.5 presents the pdf of a Laplacian distribution and the quantization intervals for a given stepsize q. As we can see, the higher the amplitude of $\tilde{y}_i$ is the lower the probability $p(\tilde{y}_i)$ is.

In order to simplify the followings notations we denote the probability $p(\tilde{y}_i)$ by $p(\tilde{y})$. Also, without loss of generalities, the quantization stepsize will be q/σ.

6.3.2 Estimation of bit planes probabilities

Three methods to estimate the binary bit plane probabilities are presented:

- Binary probabilities estimation for each bit plane $\mathbf{P}_k$.
- Contextual binary probabilities estimation depending on bits already decoded at the same position i (those bits are giving the context).
- Contextual binary probabilities estimation depending of a limited context number set to two (significant or not).

With those three estimation methods we have three coding techniques. The adaptive arithmetic coding used in bit plane coding will be improved by the knowledge of probability tables.

6.3.2.1 Adaptive arithmetic coding with model-based initialization of binary probabilities tables

In this case the initialization of probability tables is based on the generalized Gaussian modeling of variables X. So the initialization of tables will not be $p(0) = p(1)$, but it will be linked to the distribution of the source.

Calculation of binary probabilities

The probability of having the kth bit of the binary decomposition of $\tilde{y}$ equal to zero is given by:

$$\boxed{p(\mathbf{B}_k(|\tilde{y}|) = 0) = p(\tilde{y}) \times \delta_{\mathbf{B}_k(|\tilde{y}|),0}} \tag{6.15}$$

with

$$\delta_{x,y} = \begin{cases} 1 & \text{if } x = y \\ 0 & \text{if } x \neq y \end{cases}$$

The probability of having zero in the bit plane $\mathbf{P}_k$ is given by:

$$p(b_k = 0 | \, |\tilde{y}| \leq M) = \frac{p(b_k = 0, |\tilde{y}| \leq M)}{p(|\tilde{y}| \leq M)} \tag{6.16}$$

where b_k and M are respectively a random variable representing any bit in $\mathbf{P}_k$ and the higher integer we can have on K bit planes, $M = 2^K - 1$.

The probability is dependent of the number of bit plane K and so, to the number of coded integers. We supposed that the number of bit plane is transmitted to the decoder, and so it is available at the coder and the decoder.

Generalized Gaussian model is infinite and the uniform scalar quantizer is finite and determined by the number of bit planes K. So, the probability of having $|\tilde{y}| \leq M$ is not equal to one and is the sum of probabilities of variables $\tilde{y}$ with $|\tilde{y}| \leq M$:

$$p(|\tilde{y}| \leq M) = \sum_{\tilde{y}=-M}^{M} p(\tilde{y}) \tag{6.17}$$

The probability of having $b_k = 0$ and $|\tilde{y}| \leq M$ is the sum of probabilities of variables $\mathbf{B}_k(|\tilde{y}| = 0$ with $|\tilde{y}| \leq M$:

$$p(b_k = 0, |\tilde{y}| \leq M) = \sum_{\tilde{y}=-M}^{M} p(\mathbf{B}_k(|\tilde{y}|) = 0) \tag{6.18}$$

In order to simplify the followings notations we denote the probability $p(b_k = 0 | |\tilde{y}| \leq M)$ by $p_M(b_k = 0)$. So, the probability of having zero in $\mathbf{P}_k$ is given by:

$$\boxed{p_M(b_k = 0) = \frac{1}{\sum_{\tilde{y}=-M}^{M} p(\tilde{y})} \times \sum_{\tilde{y}=-M}^{M} p(\tilde{y}) \times \delta_{\mathbf{B}_k(|\tilde{y}|),0}} \tag{6.19}$$

The probability $p_M(b_k = 1)$ is given by the following relationship:

$$p_M(b_k = 1) + p_M(b_k = 0) = 1 \tag{6.20}$$

Example

Y=		-2	+5	-3	-6	+7	+3	0	+1	
		1	0	1	1	0	0	0	0	→ Bit plane of signs
K=3	2^2	0	1	0	1	1	0	0	0	→ P_2=MSB
	2^1	1	0	1	1	1	1	0	0	
	2^0	0	1	1	0	1	1	0	1	→ P_0=LSB

Figure 6.6: Example of binary estimation for each bit plane $\mathbf{P}_k$.

Fig. 6.6 presents an example of bit plane mapping of an integer sequence $\tilde{Y} = [-2, +5, -3, -6, +7, +3, 0, +1]$ with $K = 3$ bit planes. Integers with binary decomposition on $\mathbf{P}_2$ (MSB) equal to zero are -2, -3,+3, 0, +1 (values in red boxes). So the probability of having zero on $\mathbf{P}_2$ is given by:

$$p_M(b_2 = 0) = \frac{p(-2) + p(-3) + p(+3) + p(0) + p(+1)}{\sum_{\tilde{y}=-7}^{\tilde{y}=7} p(\tilde{y})} \tag{6.21}$$

In the same way, the probability of having zero on $\mathbf{P}_0$ is:

$$p_M(b_2 = 0) = \frac{p(+5) + p(0) + p(+1)}{\sum_{\tilde{y}=-7}^{\tilde{y}=7} p(\tilde{y})} \tag{6.22}$$

6.3.2.2 Model-based contextual arithmetic coding with conditional probabilities

In this case we are going to exploit the mutual information between bit planes $\mathbf{P}_k$, instead of coding them independently from the other one. We proposed a method in order to use the knowledge of the bits already decoded at the same position.

In this second method, the bit plane MSB is coded as in the first method presented just before, independent from others bit planes and with model-based initialization of probability tables. Otherwise, for the bit plane coding of $\mathbf{P}_k$ with $k < K - 1$ we used the knowledge of the previous bit planes $\mathbf{P}_{K-1} \ldots \mathbf{P}_{k+1}$.

Calculation of the probability of the context $c_k(|\tilde{y}|)$

By context $c_k(|\tilde{y}|)$, we defined the binary decomposition of y in bit planes before $\mathbf{P}_k$ ordered from LSB to MSB. For example, the binary decomposition of 6 is 011 from LSB to MSB and the context of the LSB (0) will be 11.

For every bit plane expect the MSB ($k < K-1$), the context of the binary decomposition of y, $c_k(|\tilde{y}|)$, is defined as:

$$c_k(|\tilde{y}|) = \sum_{j=k+1}^{K-1} \mathbf{B}_j(|\tilde{y}|)2^j \quad |\tilde{y}| < M \quad \forall k < K \tag{6.23}$$

The number of contexts in $\mathbf{P}_k$ is $N = 2^{K-k-1}$. We defined $\mathbf{C}_k$ has a vector containing all the possible contexts in $\mathbf{P}_k$:

$$\mathbf{C}_k = [c_k(0), \ldots, c_k(n), \ldots, c_k(N-1)] \tag{6.24}$$

For example, if we have $K = 3$ bit planes and if we are in $\mathbf{P}_1$, we have 4 contexts and $\mathbf{C}_1 = [00, 01, 10, 11]$.

The probability of having the context $c_k(|\tilde{y}|)$ in $\mathbf{P}_k$ equal to $c_k(n)$ is given by:

$$\boxed{p\left(c_k(|\tilde{y}|) = c_k(n)\right) = p\left(\tilde{y}\right) \times \prod_{j=k+1}^{K-1} \delta_{\mathbf{B}_j(|\tilde{y}|),\mathbf{B}_j(n)}} \tag{6.25}$$

We are checking that the binary decomposition of $|\tilde{y}|$ and n are of the same on previous bit planes $\mathbf{P}_{K-1} \ldots \mathbf{P}_{k+1}$. If it is the same then $p\left(c_k(|\tilde{y}|) = c_k(n)\right) = p\left(\tilde{y}\right)$ otherwise $p\left(c_k(|\tilde{y}|) = c_k(n)\right) = 0$.

Calculation of the conditional probability

We considered the a priori knowledge of the context (bit planes of rank k to $K-1$) to code $\mathbf{P}_k$. The conditional probability of having the value zero on $\mathbf{P}_k$ with knowing the context $c_k(n)$ for $k < K-1$ is defined as:

$$\boxed{p_M\left(b_k = 0 | c_k = c_k(n)\right) = \frac{p_M\left(b_k = 0, c_k = c_k(n)\right)}{p_M\left(c_k = c_k(n)\right)}} \tag{6.26}$$

where b_k and c_k are respectively a random variable representing any bit in $\mathbf{P}_k$ and a random variable representing any context in $\mathbf{P}_k$.

The probability of having the context c_k equal to $c_k(n)$ and $|\tilde{y}| \leq M$ is given by the relationship:

$$p_M\left(c_k = c_k(n)\right) = \frac{p\left(c_k = c_k(n), |\tilde{y}| \leq M\right)}{p(|\tilde{y}| \leq M)} \tag{6.27}$$

where $p(|\tilde{y}| \leq M) = \sum_{\tilde{y}=-M}^{M} p(\tilde{y})$. The probability of having $c_k = c_k(n)$ and $|\tilde{y}| \leq M$

is given by:

$$\begin{aligned} p\left(c_k = c_k(n), |\tilde{y}| \leq M\right) &= \sum_{\tilde{y}=-M}^{M} p(c_k(|\tilde{y}|) = c_k(n)) \\ &= \sum_{\tilde{y}=-M}^{M} \left[p(\tilde{y}) \times \prod_{j=k+1}^{K-1} \delta_{\mathbf{B}_j(|\tilde{y}|),\mathbf{B}_j(n)} \right] \end{aligned} \tag{6.28}$$

So we have the relationship:

$$\boxed{p_M\left(c_k = c_{k,n}\right) = \frac{1}{\sum_{\tilde{y}=-M}^{M} p(y)} \times \sum_{\tilde{y}=-M}^{M} \left[p\left(\tilde{y}\right) \times \prod_{j=k+1}^{K-1} \delta_{\mathbf{B}_j(|\tilde{y}|),\mathbf{B}_j(n)} \right]} \tag{6.29}$$

Furthermore, the probability $p_M\left(b_k = 0, c_k = c_{k,n}\right)$ for $k < K-1$ is defined as:

$$p_M\left(b_k = 0, c_k = c_k(n)\right) = \frac{p\left(b_k = 0, c_k = c_k(n)\right| |\tilde{y}| \leq M)}{p(|\tilde{y}| \leq M)} \tag{6.30}$$

where $p(|\tilde{y}| \leq M) = \sum_{\tilde{y}=-M}^{M} p(\tilde{y})$. The probability of having $b_k = 0$ with $c_k = c_k(n)$ and $|\tilde{y}| \leq M$ is given by:

$$\begin{aligned} p\left(b_k = 0, c_k = c_k(n)\right| |\tilde{y}| \leq M) &= \sum_{\tilde{y}=-M}^{M} p(b_k = 0) \times p(c_k(|\tilde{y}|) = c_k(n)) \\ &= \sum_{\tilde{y}=-M}^{M} \left[p(\tilde{y}) \times \delta_{\mathbf{B}_k(|\tilde{y}|),0} \times \prod_{j=k+1}^{K-1} \delta_{\mathbf{B}_j(|\tilde{y}|),\mathbf{B}_j(n)} \right] \end{aligned} \tag{6.31}$$

So we have the relationship:

$$\boxed{p_M\left(b_k = 0, c_k = c_k(n)\right) = \frac{1}{\sum_{\tilde{y}=-M}^{M} p(\tilde{y})} \times \sum_{\tilde{y}=-M}^{M} \left[p(\tilde{y}) \times \delta_{\mathbf{B}_k(|\tilde{y}|),0} \times \prod_{j=k+1}^{K-1} \delta_{\mathbf{B}_j(|\tilde{y}|),\mathbf{B}_j(n)} \right]} \tag{6.32}$$

Then the conditional probability of having the value zero on $\mathbf{P}_k$ with knowing the context $c_k(n)$ by using Eq. 6.29 and 6.32 is defined as:

$$\boxed{p_M\left(b_k = 0 | c_k = c_k(n)\right) = \frac{\sum_{\tilde{y}=-M}^{M} \left[p\left(\tilde{y}\right) \times \delta_{\mathbf{B}_k(|\tilde{y}|),0} \times \prod_{j=k+1}^{K-1} \delta_{\mathbf{B}_j(|\tilde{y}|),\mathbf{B}_j(n)} \right]}{\sum_{\tilde{y}=-M}^{M} \left[p\left(\tilde{y}\right) \times \prod_{j=k+1}^{K-1} \delta_{\mathbf{B}_j(|\tilde{y}|),\mathbf{B}_j(n)} \right]}} \tag{6.33}$$

In order to have all the probability we are using the following relationship:

$$\begin{cases} p_M\left(b_k = 0 | c_k = c_k(n)\right) + p_M\left(b_k = 1 | c_k = c_k(n)\right) = 1 \\ \sum_{n=0}^{N-1} p\left(c_k = c_k(n)\right) = 1 \end{cases} \tag{6.34}$$

Example

Y=	-2	+5	-3	-6	+7	+3	0	+1	
	1	0	1	1	0	0	0	0	→ Bit plane of signs
2^2	0	1	0	1	1	0	0	0	→ P_2=MSB
K=3 2^1	1	0	1	1	1	1	0	0	
2^0	0	1	1	0	1	1	0	1	→ P_0=LSB

Figure 6.7: Example of conditional probability for each bit plane $\mathbf{P}_k$.

Fig. 6.7 presents an example of bit plane mapping of an integer sequence $\tilde{Y} = [-2, +5, -3, -6, +7, +3, 0, +1]$ with $K = 3$ bit planes. For the bit plane $\mathbf{P}_1$, the vector of contexts $\mathbf{C}_1 = [0, 1]$, and for the bit plane $\mathbf{P}_0$, the vector of contexts $\mathbf{C}_1 = [00, 01, 10, 11]$.

For the bit plane $\mathbf{P}_0$, the integers with context $c_0(0) = 00$ are 0 and 1 (values in dashed red boxes). So, the probability of having the context $c_0(0) = 00$ is given by:

$$p_M(c_0 = 00) = p(1) + p(0) \tag{6.35}$$

The only integer with a binary value equal to 0 in $\mathbf{P}_0$ and context $c_0(0) = 00$ is 0. So, the probability of having a bit equal to zero in the bit plane $\mathbf{P}_0$ with $c_0(0) = 00$ is given by:

$$p_M(b_0 = 0 | c_0 = 00) = \frac{p(0)}{p(1) + p(0)} \tag{6.36}$$

6.3.2.3 Model-based contextual arithmetic coding with conditional probabilities limited to two contexts

In this third coding method, we are in the case of conditional probability as described in the second method. But, instead of having a number of contexts which is growing with the number of bit planes, we fixed to 2 the number of contexts. A context equal to zero will mean that all bits decoded in the same column i are equal to zero, and that the current coefficient is considered as non significant for the moment. A context equal to one will mean that at least one decoded bit in the

column i was equal to one, and so the current coefficient is significant.

Calculation of the probability of the context

For every bit plane expect the MSB ($k < K-1$), we defined the context $c_k(|\tilde{y}|)$ in $\mathbf{P}_k$ as:

$$c_k(|\tilde{y}|) = \begin{cases} 1 \text{ if } \exists \mathbf{B}_j(|\tilde{y}|) = 1 \text{ for } j = k+1, \ldots, K-1 \\ 0 \text{ otherwise} \end{cases} \tag{6.37}$$

The probability of having the context $c_k(|\tilde{y}|)$ in $\mathbf{P}_k$ equal to zero is given by:

$$\boxed{p\left(c_k(|\tilde{y}|) = 0\right) = p(\tilde{y}) \times \prod_{j=k+1}^{K-1} \delta_{\mathbf{B}_j(|\tilde{y}|),0}} \tag{6.38}$$

We are checking that the binary decomposition of $\tilde{y}$ is equal to zero. If it is the case then $p\left(c_k(|\tilde{y}|) = 0\right) = p(\tilde{y})$ otherwise $p\left(c_k(|\tilde{y}|) = 0\right) = 0$.

Calculation of the conditional probability limited to two contexts

We used the a priori knowledge of the context (bit plane of rank k to $K-1$) to code $\mathbf{P}_k$. The conditional probability of having the value zero with only two contexts (significant or non-significant) in $\mathbf{P}_k$ is defined as:

$$\boxed{p_M\left(b_k = 0 | c_k = 0\right) = \frac{p_M\left(b_k = 0, c_k = 0\right)}{p_M\left(c_k = 0\right)}} \tag{6.39}$$

where b_k and c_k are respectively a random variable representing any bit in $\mathbf{P}_k$ and a random variable representing any context in $\mathbf{P}_k$.

The probability $p_M\left(c_k = 0\right)$ has already been defined in Eq. 6.29, in this case $c_k(n) = 0$:

$$p_M\left(c_k = 0\right) = \frac{1}{\sum_{\tilde{y}=-M}^{M} p(\tilde{y})} \times \sum_{\tilde{y}=-M}^{M} \left[p(\tilde{y}) \times \prod_{j=k+1}^{K-1} \delta_{\mathbf{B}_j(|\tilde{y}|)|,0} \right] \tag{6.40}$$

The probability $p_M\left(b_k = 0, c_k = 0\right)$ has already been defined in Eq. 6.32, in this case $c_k(n) = 0$:

$$p_M\left(b_k = 0, c_k = 0\right) = \frac{1}{\sum_{\tilde{y}=-M}^{M} p(\tilde{y})} \times \sum_{\tilde{y}=-M}^{M} \left[p(\tilde{y}) \times \prod_{j=k}^{K-1} \delta_{\mathbf{B}_j(|\tilde{y}|),0} \right] \tag{6.41}$$

So, the conditional probability of having the value zero with only two contexts (significant or non-significant) in $\mathbf{P}_k$ is defined as:

$$\boxed{p_M\left(b_k=0|c_k=0\right)=\frac{\sum_{\tilde{y}=-M}^{M}\left[p(\tilde{y})\times\prod_{j=k}^{K-1}\delta_{\mathbf{B}_j(|\tilde{y}|),0}\right]}{\sum_{\tilde{y}=-M}^{M}\left[p(\tilde{y})\times\prod_{j=k+1}^{K-1}\delta_{\mathbf{B}_j(|\tilde{y}|),0}\right]}} \tag{6.42}$$

In order to have all the probability we are using the following relationship:

$$\begin{cases} p_M\left(b_k=0|c_k=0\right)+p_M\left(b_k=1|c_k=0\right)=1 \\ p\left(c_k=0\right)+p\left(c_k=1\right)=1 \end{cases} \tag{6.43}$$

Example

Y=	-2	+5	-3	-6	+7	+3	0	+1	
	1	0	1	1	0	0	0	0	→ Bit plane of signs
2^2	0	1	0	1	1	0	0	0	→ P_2=MSB
K=3 2^1	1	0	1	1	1	1	0	0	
2^0	0	1	1	0	1	1	0	1	→ P_0=LSB

Figure 6.8: Example of conditional probability with only two contexts for each bit plane $\mathbf{P}_k$.

Fig. 6.8 presents an example of bit plane mapping of an integer sequence $\tilde{Y}=[-2,+5,-3,-6,+7,+3,0,+1]$ with $K=3$ bit planes. For the bit plane $\mathbf{P}_1$, variables with context $c_1=0$ - having 0 on every bit plan before the current one (on $\mathbf{P}_2$ in this example) are -2, -3, +3, 0, +1. So, the probability of having a context equal to zero on $\mathbf{P}_1$ is given by:

$$p_M\left(c_1=0\right)=p(-2)+p(-3)+p(3)+p(0)+p(1) \tag{6.44}$$

For $\mathbf{P}_0$, in order to know if a context is zero, we must check bit planes $\mathbf{P}_1$ and $\mathbf{P}_2$. The only variables who have all bits equal to zero on $\mathbf{P}_1$ and $\mathbf{P}_2$ are 0 and +1. So, the probability of having $c_0=0$ on $\mathbf{P}_0$ is given by:

$$p_M\left(c_0=0\right)=p(0)+p(1) \tag{6.45}$$

The probability of having $b_0=0$ with $c_0=0$ in $\mathbf{P}_0$ is given by:

$$p_M\left(b_0=0|c_0=0\right)=\frac{p(0)}{p(0)+p(1)} \tag{6.46}$$

6.4 Transform Audio Coding with Model-Based Bit Plane Coding

6.4.1 Encoder

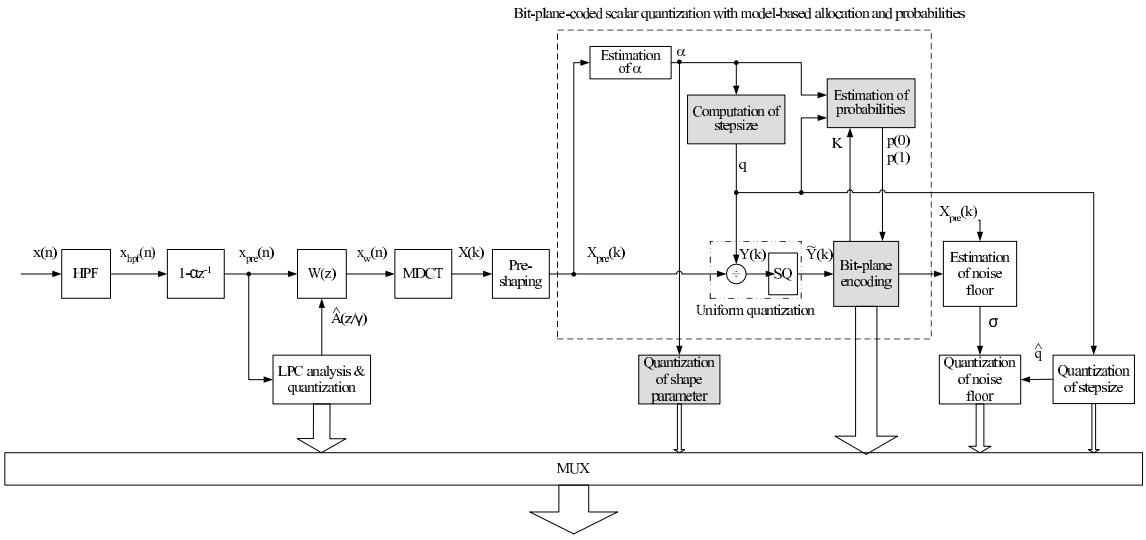


Figure 6.9: Block diagram of the proposed bit plane transform encoder.

The proposed encoder is illustrated in Fig. 6.9. The encoder employs a linear-predictive weighting filter followed by MDCT coding. The input sampling frequency is 16000 Hz, while the frame length is 20 ms with a lookahead of 25 ms. The effective bandwidth of the input signal is considered to be 50-7000 Hz. The analysis done on the input signal $x(n)$ is the same than for the Stack-Run encoder presented in Section 5.2.

A generalized Gaussian model approximates the distribution of the spectrum $X_{pre}(k)$ composed of $N = 320$ coefficients. Mallat's method [69] is used to estimate the shape parameter α (see also Section 3.2.2). The pre-shaped spectrum $X_{pre}(k)$ is divided by stepsize q and the resulting coefficients $Y(k)$ are encoded by uniform scalar quantization. For a given spectrum $Y(k)$ the spectrum $\tilde{Y}(k)$ after quantization is defined as:

$$\tilde{Y}(k) = [Y(k)] = \left[\frac{X_{pre}(k)}{q}\right] \tag{6.47}$$

Only the first 280 coefficients of the spectrum $Y(k)$ corresponding to the 0-7000 Hz band are coded; the last 40 coefficients are discarded. The integer sequence $\tilde{Y}(k)$ is encoded by bit plane coding.

Here the stepsize q is set based on the asymptotic estimation [88]:

$$q = q_{opt} \times 2^{-margin} = \sqrt{\frac{6\lambda_{opt}}{\ln(2)}} \times 2^{-margin} \tag{6.48}$$

where $margin$ is a value chosen to ensure that the encoder will always use the whole bit budget, and λ_{opt} is given by :

$$\lambda_{opt} = 2\ln(2)h\sigma^2 \sum_{k=1}^{N} 2^{-2b_k} \tag{6.49}$$

where σ and h are respectively the standard deviation and a function of the pdf of $X_{pre}(k)$ given by [88], b_k is the number of bits per sample to code $X_{pre}(k)$ and $margin = 2$.

In this work bit planes are coded using adaptive arithmetic coding [126, 62]. Before using bit plane coding, the probabilities of 0 and 1 in each bit plane are needed. We are using the knowledge of the model parameters σ, α and stepsize q to estimate efficiently those probabilities following one of the three methods presented in section 6.3. Then a noise level estimation is performed on the spectrum $Y(k)$ after bit plane coding. The noise floor σ is estimated as:

$$\sigma = r.m.s. \{X_{pre}(k) \,|\, Y(k) = 0\} \tag{6.50}$$

with the additional constraint that $Y(k)$ must belong to a long zero run to be really considered in the above r.m.s. calculation. The step size q is scalar quantized in log domain with 7 bits. The shape parameter α is scalar quantized with 3 bits and the number of bit plane K is scalar quantized with 4 bits.

6.4.2 Bit allocation

Static distributions of bits to parameters

The parameters of the proposed coder are the Line Spectrum Frequency (LSF) parameters, the step size, the shape parameter and the noise floor level. The bit allocation to the parameters is detailed in Table 6.1, where B_{tot} is the total number of bits per frame. For instance at 24 kbit/s, $B_{tot} = 480$ bits. The allocation (in bits per sample) to stack-run coding is $B = (B_{tot} - 63)/280$, where 280 is the number of coefficients in each frame and $63(40 + 7 + 9 + 3 + 4)$ is the number of bits needed to transmit the parameters to the decoder.

Rate control for bit plane coding

Note that the encoding stops when the bit budget is reached; all the non-coded bits in bit planes are replaced by zero. Therefore there is no need to implement a rate control procedure, unlike [79] (see also Chapter 5).

Table 6.1: Bit allocation for the bit plane transform audio coding.

Parameter	Number of bits
LSF	40
Step size q	7
Noise floor σ	9
Shape parameter α	3
Number of bit plane K	4
Bit plane coding	B_{tot}-63
Total	B_{tot}

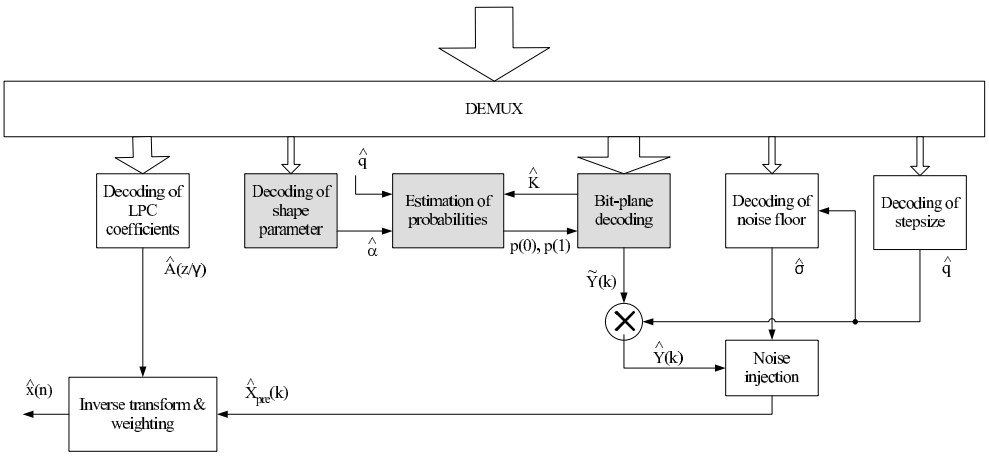


Figure 6.10: Block diagram of the proposed bit plane transform decoder.

6.4.3 Decoder

The decoder in error-free conditions is illustrated in Fig. 6.10. The bit plane decoder is equivalent to the Stack-Run decoder presented in Section 5.2. An estimation of probabilities for the bit plane decoding is done based on the knowledge of the decoded shape parameter $\hat{\alpha}$, the decoded stepsize $\hat{q}$ and the decoded number of bit plane $\hat{K}$.

6.5 Experimental Results and Discussion

A database of 24 clean speech samples in French language (6 male and female speakers×4 sentence-pairs) and 16 clean music samples (4 types×4 samples) of 8 seconds is used. These samples are sampled at 16 kHz, preprocessed by the P.341 filter of ITU-T G.191A and normalized to -26 dB_{ov} using the P.56 speech voltmeter.

6.5.1 Objective quality results

WB-PESQ [84] is used to evaluate the quality of the proposed coder and compare it with ITU-T G.722.1 and Stack-run coding. Only clean speech samples are used to compute the average WB-PESQ scores at various bitrates. The bit rate varies from 16 to 40 kbit/s with a step of 4 kbit/s for our coder. ITU-T G722.1 is tested at 24 and 32 kbit/s.

Fig. 6.11 shows the WB-PESQ score obtains for the three coders. Theses results suggest that the quality of the proposed scalable bit plane coder is better than ITU-T G.722.1 at 32 kbit/s (0.2 MOS-LQ0 difference). It also suggest that using model-based initialization for the arithmetic coder improves the performance of the coder especially at low bitrate (0.2-0.1 MOS-LQ0 difference). So if we are considering the proposed scalable bit plane coder with model-based initialization of the arithmetic coder, its quality is better than ITU-T G.722.1 at 24 and 32 kbit/s (0.1-0.3 MOS-LQ0 difference). Still the performance of the bit plane coder is under stack-run coding especially at low bitrate. The reason is that stack-run coding is more efficient than bit plane coding on long sequences of zeros which is the case at low bitrate.

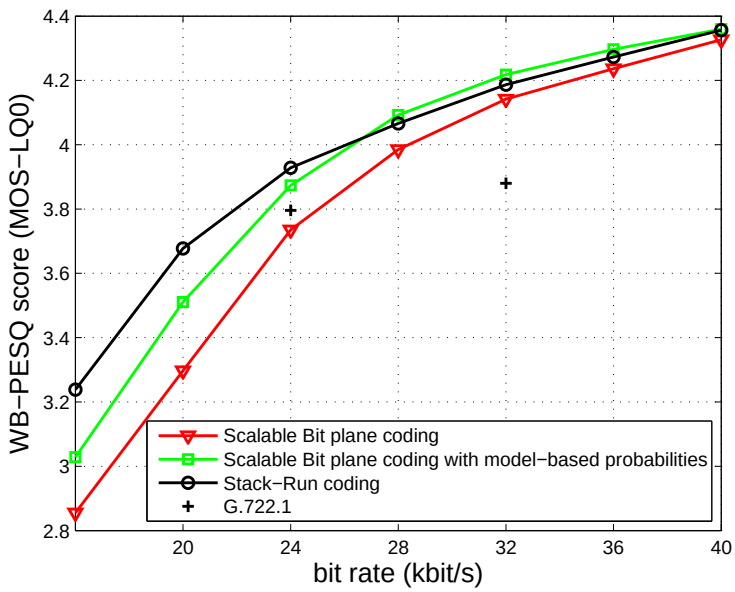


Figure 6.11: Average wideband PESQ score (without noise injection).

Fig. 6.12 shows the difference between a bit plane coder with encoding and decoding at the same bitrate and a scalable coder in the sense that the decoding rate is smaller that the encoding one. The WB-PESQ score suggest that the scalable and the non-scalable coders have the same performance.

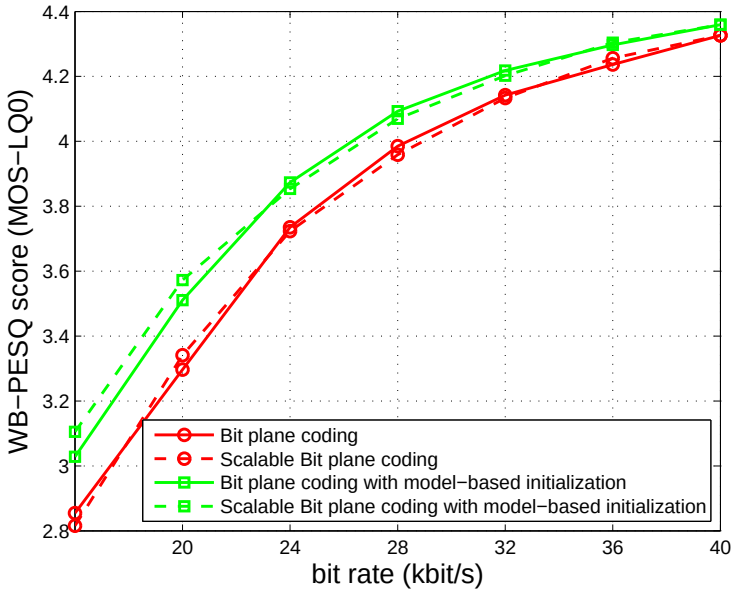


Figure 6.12: Comparison of bit plane coding with encoding/decoding at the same bitrate or different ones (without noise injection).

6.5.2 Subjective quality results

Subjective tests at 32 kbit/s have been conducted: one for speech, another for music. At 32 kbit/s the proposed coder is equivalent to reference coders in both cases (G.722.1 and stack-run coding). Informal listening confirmed the quality difference from 16 to 32 kbit/s between the proposed coder and stack-run coding, predicted by WB-PESQ. Note that WB-PESQ is revelant in this latter case, as the proposed coder and stack-run coder have very close coding structures (only MDCT quantization methods differ). Furthermore we still have to improve the noise injection of the proposed coder at low bitrate in order to compare it with the reference coders.

6.5.3 Complexity

This embedded transform audio coder has been implemented in language C. The algorithmic delay of the proposed embedded coder and stack-run coding is 45 ms (20 ms for the frame, 20ms for the MDCT and 5 ms for the lookahead), while that of G.722.1 is 40 ms.
The computational complexity of G.722.1 is low which is also the case for the proposed embedded coder since rate control is automatically handled by bit plane coding.
The memory requirements (in terms of data ROM) for the proposed coder consists mainly of the storage of GMM parameters for LPC quantization and MDCT computation tables.

6.6 Conclusion

In this chapter we proposed an embedded speech and audio coder based on generalized Gaussian modeling and bit plane coding. This coder was compared against ITU-T G.722.1 and stack-run audio coding. The generalized Gaussian model allows to estimate efficiently symbol probability in bit planes. This model-based approach brings an improvement of 0.1-0.4 MOS-LQ0 compared with a baseline bit plane coder. The proposed coder reaches a performance similar to non-embedded coding such as stack-run coding or ITU-T G.722.1, which is remarkable. Further work will be focused on improving quality at low bit rates to reduce the performance penalty and to handle multiple constraints (e.g. sampling frequency, frame length, runs of zeros) of bitstream scalability.

Chapter 7

Conclusion

Conclusion générale

Dans cette thèse, nous nous sommes intéressés au codage de la parole et de l'audio. Nous avons présentés des techniques de codage permettant d'avoir des performances meilleures ou équivalentes à celle des codeurs actuels tout en pouvant s'adapter en temps-réel à différentes contraintes (débit, largeur de bande, retard, etc.).

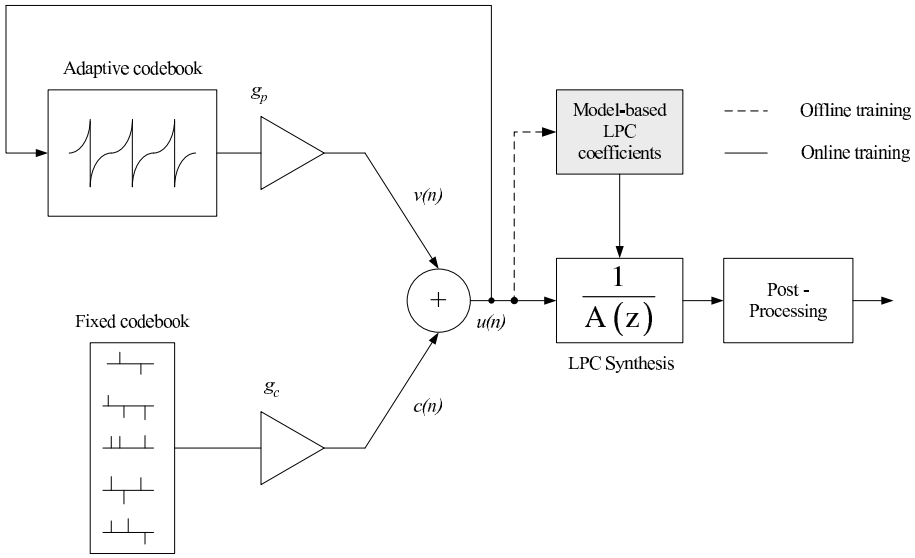


Figure 7.1: Schéma de l'encodage CELP modifié en utilisant un modèle.

Dans un premier temps, nous nous sommes intéressés à la quantification des coefficients LPC. Des approches de quantification basé modèle ont déjà été proposées récemment dans [104, 111]. Ces approches consistent à modéliser les coefficients LPC par un mélange de gaussiennes. Nous avons proposé d'améliorer ces techniques en modélisant l'erreur de prédiction des coefficients LPC par un

modèle gaussien généralisé. Le quantificateur prédictif proposé est basé sur la méthode de quantification KLT avec l'utilisation du modèle Gaussien généralisé pour approcher les paramètres LPC. Les performances obtenues pour ce quantificateur sont équivalentes à celle du quantificateur utilisé dans le codeur ITU-T G.722.2 [40] en termes de distortion spectrale (SD). Par contre, le coût mémoire et la complexité de calcul du quantificateur proposé est moindre par rapport au quantificateur utilisé dans l'ITU-T G.722.2. Le schéma de quantification obtenu pourrait être utilisé dans les codeurs basés sur le modèle CELP [106]. La Fig. 7.1 présente la possibilité de modification de l'encodeur CELP en utilisant un modèle (GMM ou Gaussien généralisé). L'utilisation d'un modèle statistique long-terme (estimation du modèle sur une base d'entraînement) permet donc d'améliorer les performances et la complexité des codeurs basé CELP.

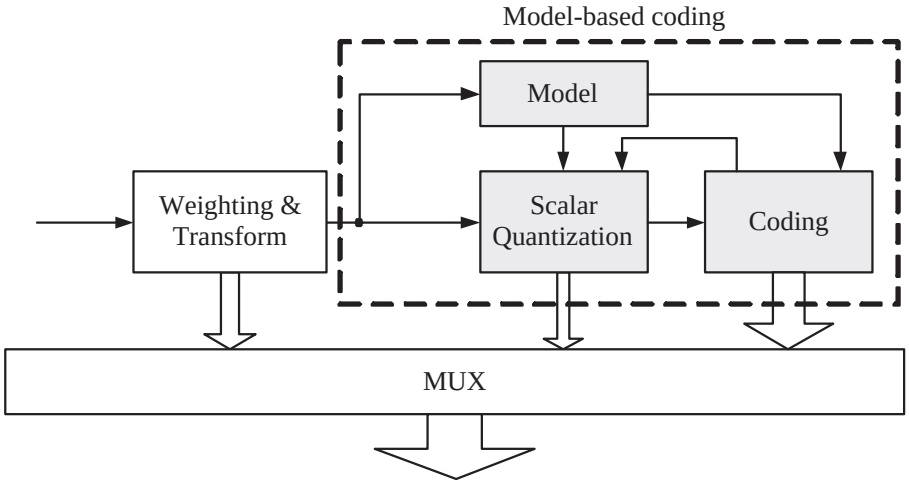


Figure 7.2: Schéma de l'encodeur par transformée, modifié en utilisant un modèle.

Dans un second temps, nous nous sommes intéressés au codage par transformée de la parole et de l'audio. Deux schémas de codage basé modèle ont été présentés, l'un utilisant le codage stack-run et l'autre le codage par plan de bits. Dans le cas du codeur avec codage stack-run, le modèle Gaussien généralisé est utilisé pour modéliser les coefficients après la transformation MDCT. Le modèle obtenu permet d'optimiser l'allocation de débit et la largeur de la zone morte du quantificateur scalaire. Pour le codeur avec codage par plan de bits, le modèle Gaussien généralisé est aussi utilisé pour modéliser les coefficients après la transformation MDCT. Le modèle obtenu sert à l'initialisation des tables de probabilités du codeur arithmétique et au calcul du pas de quantification. Les performances de ces deux schémas de codage sont comparées avec celle du codeur ITU-T G.722.1 [39]. Afin d'évaluer la qualité des codeurs, des mesures objectives et des tests subjectifs ont été menés. Pour les mesures objectives on se sert des scores WB-PESQ [84] afin de comparer la qualité des codeurs pour la parole. Les tests subjectifs sont des tests d'écoute où

l'auditeur compare la qualité des codeurs pour la parole et la musique. Plusieurs tests subjectifs ont été effectués afin de comparer le codeur de référence ITU-T G.722.1 avec le codeur avec codage stack-run et celui avec codage par plan de bits. Dans le cas des mesures objectives ou pour les tests subjectifs, la qualité des codeurs proposés est meilleure que celle du codeur ITU-T G.722.1 à bas débit et est équivalente à haut débit. La qualité du codeur avec codage stack-run est meilleure que celle avec codage par plan de bits à bas débit et elle est équivalente à haut débit. Par contre, la complexité de calcul est plus importante car il nécessite une boucle de contrôle du débit qui est coûteuse en calcul. Les schémas de codage obtenus pourraient très bien être utilisés pour modifier les codeurs par transformées [55, 54, 85]. La Fig. 7.2 présente la modification du schéma de codage par transformées avec l'utilisation du modèle Gaussien généralisé. L'utilisation d'un modèle statistique court-terme (estimation du modèle en ligne sur un nombre limité de coefficients) permet donc d'améliorer la flexibilité des codeurs par transformée en leur permettant de s'adapter en temps-réel à différentes contraintes (débit, largeur de bande, retard, etc.).

Publications and Patents

Publications

1. **M. Oger**, S. Ragot and M. Antonini, *Codage basé modèle des paramètres LPC pour la compression de la paole*, in Proc. Coresa, Rennes, France, November 2005.

2. **M. Oger**, J. Bensa, S. Ragot and M. Antonini, *Stack-Run Audio Coding*, in 120th AES convention, Preprint 6805, Paris, France, May 2006.

3. **M. Oger**, S. Ragot and M. Antonini, *Low-complexity wideband LSF quantization by predictive KLT coding and generalized Gaussian modeling*, in Proc. Eusipco, Florence, Italy, September 2006.

4. **M. Oger**, S. Ragot and M. Antonini, *Transform audio coding with arithmetic-coded scalar quantization and model-based bit allocation*, Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), p. 545-548, Hawaii, USA, April 2007.

Patents

1. S. Ragot, **M. Oger** and M. Antonini, *Codage et décodage algébrique optimisé*, FR-07 53414.

2. **M. Oger**, T. M. N. Hoang, S. Ragot and M. Antonini, *Codage/Décodage modèle plans de bits*, FR-07 06001

Publications submitted

1. **M. Oger**, S. Ragot and M. Antonini, *Model-based deadzone optimization for stack-run coding with uniform scalar quantization*, Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Las Vegas, USA, April 2008.

2. T. M. N. Hoang, **M. Oger**, S. Ragot and M. Antonini, *Embedded transform coding of audio signals by model-based bit plane coding*, Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Las Vegas, USA, April 2008.

Publications to be submitted

1. **M. Oger**, T. M. N. Hoang , S. Ragot and M. Antonini, *Flexible embedded transform coding of audio signals by model-based bit plane coding*, will be submitted to IEEE Trans. on Signal Processing.

Bibliography

[1] ISO/IEC 11172-3. *Coding of moving pictures and associated audio for digital storage media at up to 1.5 Mbit/s-audio part.* Nov. 1993.

[2] ISO/IEC 13818-3. *Information technology - Generic coding of moving pictures and associated audio information - Part 3: Audio.* 1998.

[3] ISO/IEC 13818-7. *Information technology - Generic coding of moving pictures and associated audio information - Part 7: Advanced Audio Coding.* 2003.

[4] ISO/IEC 14496-3. *Information technology - Coding of Audio-visual objects - Part 3: Audio.* 2001.

[5] ISO/IEC 14496-3/Amd.1. *Information technology - Coding of Audio-visual objects - Part 3: Audio - Amd.1: Bandwidth extension.* 2003.

[6] ISO/IEC 14496-3/Amd.2. *Information technology - Coding of Audio-visual objects - Part 3: Audio - Amd.2: Parametric coding for high-quality audio.* 2004.

[7] 3GPP TS 26.290. *Audio codec processing functions; Extended Adaptive Multi-Rate - Wideband (AMR-WB+) codec; Transcoding functions.* 2005.

[8] 3GPP TS 26.403. *Enhanced aacPlus general audio codec; Encoder specification AAC part.* 2006.

[9] J. P. Adoul and C. Lamblin. A comparison of some algabraic structures for CELP coding of speech. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1953–1956, 1987.

[10] B. S. Atal and M. R. Schroeder. Predictive coding of speech and subjective error criteria. *IEEE Trans. Acoust. Speech, Signal Processing*, ASSP-27:247–254, jun 1979.

[11] B. Bessette. Methods and devices for low-frequency emphasis during audio compression based on acelp/tcx.

[12] B. Bessette, R. Salami, R. Lefebvre, M. Jelinek, J. Rotola-Pukkila, J. Vainio, H. Mikkola, and K. Järvinen. The Adaptative Multirate Wideband speech codec (AMR-WB). *IEEE Trans. Speech and Audio Proc.*, 10(8):620–636, 2002.

[13] M. Bosi, K. Brandenburg, S. Quakenbush, L. Fielder, K. Akagiri, H. Fuchs, M. Dietz, J. Herre, G. davidson, and Y. Oikawa. Iso/iec mpeg-2 Advanced Audio Coding. *Journal of the Audio Engineering Society*, 45(10):789–814, 1997.

[14] M. Bosi and R. E. Goldberg. *Introduction to digital audio coding and standards.* Kluwer Academic Publishers, 2003.

[15] K. Brandenburg. MP3 and AAC explained. *Presented at the AES 17th International Conference on High Quality Audio Coding*, page Preprint 23, 1999.

[16] J. Breebart, S. van de Par, A. Kohlrausch, and E. Schuijers. High-quality parametric spatial audio coding at low bit rates. *Presented at the AES 116th Convention*, page Preprint 6072, 2004.

[17] J. Breebart, S. van de Par, A. Kohlrausch, and E. Schuijers. Parametric coding of stereo audio. *Journal Appl. Signal Process.*, 9:1305–1322, 2005.

[18] ITU-R Rec BS.1116. *Methods for the subjective assessment of small impairments in audio systems including multichannel sound systems.* Oct 1997.

[19] ITU-R Rec BS.1387. *Method for objective measurements of perceived audio quality.* Nov 2001.

[20] ITU-R Rec BS.1534. *Method for the subjective assessment of intermediate quality levels of coding systems.* Jan 2003.

[21] ITU-T G.722.1 Annex C. *Coding at 24 kbit/s and 32 kbit/s for Hand-free Operations in Systems with Low Frame Loss-Annex C.* 2004.

[22] P. C. Chang and R. M. Gray. Gradient algorithmes for designing predictive vector quantizers. *IEEE Trans. on Acoustics Speech and Signal Proc.*, ASSP-34:679–690, Aug 1986.

[23] J. H. Chen and A. Gersho. Adaptive postfiltering for quality enhancement of coded speech. *IEEE Trans. on Speech and Audio Proc.*, 3:59–71, jan 1995.

[24] Y. M. Cheng, D. O'Shaughnessy, and P. Mermelstein. Statisical recovery of wideband speech from narrowband speech. *IEEE Trans. on Speech and Audio Proc.*, 2:544–548, 1992.

[25] P. Combescure and al. A 16, 24, 32 kbit/s wideband speech coding based on acelp. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1:5–8, Mar 1999.

[26] A. Crossman. A variable bit rate audio coder for videoconferencing. *IEEE workshop on Speech and Coding for Telecommunication*, pages 29–30, 1993.

[27] A. Crossman. A fixed bit rate wideband codec for uw in isdn videotelephony. *IEEE workshop on Speech and Coding for Telecommunication*, pages 29–30, 1994.

[28] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the em algorithm. *JRSSB*, 39(1):1–38, 1977.

[29] M. Dietz, L. Liljeryd, K. Kjörling, and O. Kunz. Spectral Band Replication, a novel approach in audio coding. *Presented at the AES 112th Convention*, page Preprint 5553, May 2002.

[30] P. Duhamel, Y. Mahieux, and J. P. Petit. A fast algorithm for the implementation of filter bank based on time domain aliasing cancellation. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2209–2212, May 1991.

[31] C. Dunn. Efficient audio coding with fine-grain scalability. *Presented at the AES 111th Convention*, page Preprint 5492, 2001.

[32] P. Ekstrand. Bandwith extension of audio signals by Spectral Band Replication. *Proc. IEEE Benelux Workshop on Model based Processing and Coding Audio (MPCA-2002)*, pages 53–58, Nov 2002.

[33] D. Esteban and C. Galand. Application of quadrature mirror filters to split band voice coding schemes. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 191–195, 1977.

[34] C. Faller and F. Baumgarte. Estimation of auditory spatial cues for binaural cue coding. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2:1801–1804, 2002.

[35] ITU-T Rec G.108. *Application of the E-model: A planning guide.* Sept 1999.

[36] ITU-T Rec G.191. *Software tools for speech and audio coding standardization.* Sept 2005.

[37] ITU-T G.711. *Pulse code modulation (PCM) of voice frequencies.* 1988.

[38] ITU-T G.722. *7 kHz audio-coding within 64 kbit/s.* 1988.

[39] ITU-T G.722.1. *Coding at 24 kbit/s and 32 kbit/s for Hand-free Operations in Systems with Low Frame Loss.* 1999.

[40] ITU-T G.722.2. *Wideband Coding of Speech at Around 16 kbit/s using Adaptative Multirate Wideband.* 2002.

[41] ITU-T G.729.1. *G.729 based Embedded Variable bit-rate coder: An 8-32 kbit/s scalable wideband coder bitstream interoperable with G.729*. 2006.

[42] C. R. Galand and H. J. Nussbaumer. New quadrature mirror filter structures. *IEEE Trans. Acoust. speech Signal Proc.*, ASP-32(3):522–530, 1984.

[43] A. Gersho. Principles of quantization. *IEEE Trans. on Circuit and systems*, 25(7):427–436, jul. 1978.

[44] A. Gersho and V. Cuperman. Vector quantization: a pattern-matching technique for speech coding. *IEEE Communications Magazine*, 21:15–21, Dec. 1983.

[45] A. Gersho and R. M. Gray. *Vector Quantization and Signal Compression*. Kluwer Academic Publishers, 1993.

[46] P. Hedelin and J. Skoglund. Vector quantization based on Gaussian mixture models. *IEEE Trans. Speech and Audio Proc.*, 8(4):385–401, Jul. 2000.

[47] J. Herre, K. Brandenburg, and D. Lederer. Intensity stereo coding. *Presented at the 96th AES Convention*, page Preprint 3799, 1994.

[48] Y. Hiwasaki, T. Mori, H. Ohmuro, J. Ikedo, D. Tokumoto, and A. Kataoka. Scalable speech coding technology for high-quality ubiquitous communications. *NTT technical review*, 2(3):53–58, mar 2004.

[49] P. G. Howard and J. S. Vitter. Arithmetic coding for data compression. *Proceedings of the IEEE*, 82(6):857–865, june 1994.

[50] F. Itakura. Line spectrum representation of linear predictive coefficients of speech signals. *J. Acoust. Soc. Am.*, 57, apr. 1975.

[51] P. Jax. *Enhancement of bandlimited speech signals: algorithms and theoritical bounds*. PhD thesis, Aachener Beiträge zu Digitalen Nachrichtensystemen (ABDN), 2002.

[52] P. Jax and P. Vary. Bandwidth extension of speech signals: a catalyst for the introduction of wideband speech coding. *IEEE Communications Magazine*, pages 106–111, May 2006.

[53] N. S. Jayant and P. Noll. *Digital Coding of Waveforms*. Prentice-Hall,Inc, 1984.

[54] J. D. Johnston. Perceptual transform coding of wideband stereo signals. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2524–2527, May 1988.

[55] 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.

[56] J. D. Johnston. Estimation of perceptual entropy using noise masking criteria. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1993–1996, May 1989.

[57] J. D. Johnston and A. J. Ferreira. Sum-difference stereo transform coding. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 23–26, Mar 1992.

[58] R. L. Joshi and T. R. Fisher. Comparison of generalized Gaussian and Laplacian modeling in DCT image coding. *IEEE. Sign. Proc. Lett.*, 2(5):81–82, May 1995.

[59] S. W. Kim, S. H. Park, and Y. B. Kim. Fine grain scalability in mpeg-4 audio. *Presented at the AES 111th Convention*, page Preprint 5491.

[60] K. Kokkinakis and A. K. Nandi. Speech modelling based on generalized Gaussian probability density functions. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 381–384, 2005.

[61] S. Kullback. *Information Theory and Statistics.* New York: Wiley, 1959.

[62] G. G. Langdon. An introduction to arithmetic coding. *IBM J. Res. Develop*, 28(2):135–149, mar 1984.

[63] R. Lefebvre, R. Salami, C. Laflamme, and J. P. Adoul. High quality codinf of wideband audio signals using transform coded excitation (TCX). *IEEE Acous. Speech and Signal Proc.*, 1:193–196, Apr 1994.

[64] J. Li. Embedded Audio Coding (eac). *ACM Multimedia 2002*, dec 2002.

[65] W. Li. overview of fine granularity scalability in MPEG-4 video standard. *IEEE Trans. Circuits Syst. Video Technol.*, 11(3):301–316, mar.

[66] X. Maitre. 7 khz audio coding within 64 kbit/s. *IEEE Journal on selected areas in Communications*, pages 283–298, February 1988.

[67] J. Makhoul. Linear prediction: a tutorial review. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 63(5):561–580, Apr 1975.

[68] J. Makhoul and M. Berouti. High-frequency regeneration in speech coding systems. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 428–431, Apr 1979.

[69] S. G. Mallat. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Trans. Patt. Anal. Machine Intell.*, 11:674–693, jul 1989.

[70] S. G. Mallat. Analysis of low bit rate image transform coding. *IEEE Trans. on Signal Proc.*, 46:1027–1042, apr 1998.

[71] H. S. Malvar. Lapped transforms for efficient transform/subbband coding. *IEEE Trans. on Acoustics, Speech and Signal Processing, vol. 38, no. 6*, pages 969–978, 1990.

[72] H. S. Malvar. *Signal Processing with lapped transforms.* Artech House Publishers, 1992.

[73] J. D. Markel and A. H. Gray. *Linear Prediction of Speech.* Springer-Verlag, 1976.

[74] J. Mäkinen, B. Bessette, S. Bruhn, P. Ojala, R. Salami, and A. Taleb. AMR-WB+: a new audio coding standard for 3 rd generation mobile audio services. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2:1109–1112, May 2005.

[75] A. Moffat, R. Neal, and I. H. Witten. Arithmetic coding revisited. *ACM Transactions on Information Systems*, 16(3):256–294, July 1998.

[76] R. M. Neal. Fast arithmetic coding using low-precision division. *Unpublished Manuscript*, 1987.

[77] M. Oger, J. Bensa, S. Ragot, and M. Antonini. Stack-run audio coding. *in 120th AES convention*, page Preprint 6805, May 2006.

[78] M. Oger, S. Ragot, and M. Antonini. Low-complexity wideband LSF quantization by predictive KLT coding and generalized gaussian modeling. *in Proc. Eusipco*, 2006.

[79] M. Oger, S. Ragot, and M. Antonini. Transform audio coding with arithmetic-coded scalar quantization and model-based bit allocation. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 4:545–548, 2007.

[80] M. Oger, S. Ragot, and R. Lefebvre. Companded lattice VQ for efficient parametric LPC quantization. *in Proc. Eusipco*, 2004.

[81] ITU-T Rec P.800. *Methods for subjective determination of transmission quality.* Aug. 1996.

[82] ITU-T Rec P.862. *Perceptual evaluation of speech quality (PESQ): An objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs.* Feb 2001.

[83] ITU-T Rec P.862.1. *Mapping function for transforming P.862 raw result scores to MOS-LQO.* Nov 2003.

[84] ITU-T Rec P.862.2. *Wideband extension to Recommendation P.862 for the assessment of wideband telephone networks and speech codecs.* Nov 2005.

[85] T. Painter and A. Spanias. Perceptual audio coding. *Proc. IEEE, vol. 88*, (4):451–513, 2000.

[86] K. K. Paliwal and W. B. Kleijn. *Speech Coding and Synthesis.* Elsevier Science, 1995.

[87] A. Papoulis. *Probability, random variables, and stochastic processes.* Mc Graw-Hill, 1965.

[88] C. Parisot. *Allocation basées modèles et transformée en ondelettes au fil de l'eau pour le codage d'images et de vidéos.* PhD thesis, Université de Nice-Sophia Antipolis, 2003.

[89] C. Parisot, M. Antonini, and M. Barlaud. 3d scan based wavelet transform and quality control for video coding. *EURASIP*, 2003(1):521–528, Jan 2003.

[90] S. H. Park, Y. B. Kim, S. W. Kim, and Y. S. Seo. Multi-layer bit-sliced bit-rate scalable audio coding. *Presented at the AES 103rd Convention*, page Preprint 4520, aug 1997.

[91] W. B. Pennebaker and J. L. Mitchell. Probability estimation for the q-coder. *IBM J. Res. Develop*, 32(6):737–752, nov 1988.

[92] W. B. Pennebaker, J. L. Mitchell, G. G. Langdon, and R. B. arps. An overview of the basic principles of the q-coder adaptive binary arithmetic coder. *IBM J. Res. Develop*, 32(6):717–726, nov 1988.

[93] ITU-T Rec P.OLQA. *Objective Listening Quality Assessment.* In progress.

[94] ITU-T Rec P.ONRA. *Objective Noise Reduction Assessment.* In progress.

[95] C. Pépin, P. Raffy, and R. M. Gray. Robust stack-run coding for low bit-rate image transmission over noisy channels. *IEEE Signal Proc. Letters*, 9(7):196–199, jul 2002.

[96] J. Princen, A. W. Johnston, and A. Bradley. Subband/transform coding using filter bank designs based on time domain cancellation. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2161–2164, 1987.

[97] S. Ragot and al. ITU-T G.729.1: an 8-32 kbit/s scalable coder interoperable with G.729 for wideband telephony and voice over IP. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 4:529–532, May 2007.

[98] S. Ragot, B. Bessette, and R. Lefebvre. Low-complexity multi-rate lattice vector quantization with application to wideband speech coding at 32 kbit/s. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1:501–504, May 2004.

[99] S. Ragot, H. Lahdili, and R. Lefebvre. Wideband LSF quantization by generalized Voronoi codes. *Proc. Eurospeech*, pages 2319–2322, Sep. 2001.

[100] R. A. Redner and H. F. Walker. Mixture densities, maximum likelihood and the em algorithm. *SIAM Rev., vol. 26, no. 2*, pages 195–239, 1984.

[101] A. Said and W. A. Pearlman. A new, fast, and efficient image codec based on set partitioning in hierarchical trees. *Proc. IEEE Trans. Circuits and sys. Video Tech.*, 6:243–250, June 1996.

[102] R. Salami, C. Laflamme, and J. P. Adoul. Acelp speech coding at 8 kbit/s with a 10 ms frame: a candidate for ccitt standardization. *Proc. IEEE Workshop on Speech Coding for Telecommunications*, pages 23–24, Oct 1993.

[103] D. Salomon. *Data Compression: The Complete Reference.* Springer, 2004.

[104] J. Samuelsson. *Towards Asymptotic Vector Quantization.* PhD thesis, School of Electrical and Computer Engineering, Chalmers University of Technology, 2001.

[105] J. Samuelsson. Waveform quantization of speech using Gaussian mixture models. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1:165–168, 2004.

[106] M. R. Schroeder and B. S. Atal. Code-Excited Linear Prediction (CELP): High-quality speech at very low bit rates. *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 937–940, 1985.

[107] E. Schuijers, J. Breebaart, H. Purnhagen, and J. Engdegård. Low complexity parametric stereo coding. *Presented at AES 114th Convention*, page Preprint 6073, May 2004.

[108] E. Schuijers, W. Oomen, B. den Brinker, and J. Breebaart. Advances in parametric coding for high-quality audio. *Presented at AES 114th Convention*, page Preprint 5852, Mar 2003.

[109] J. M. Shapiro. Embedded image coding using zerotrees of wavelet coefficients. *IEEE Trans. Signal Processing*, 41:3445–3462, Dec 1993.

[110] J. K. Su and R. M. Mersereau. Coding using Gaussian mixture and generalized Gaussian models. *Proc IEEE Int. Conf Image Processing (ICIP)*, 1:217–220, sep 1996.

[111] A. D. Subramaniam. *Gaussian mixture models in Compression and Communication.* PhD thesis, University of California, San Diego, 2003.

[112] A. D. Subramaniam, W. R. Gardner, and B. D. Rao. Low-complexity source coding using Gaussian mixture models, lattice vector quantization, and recursive coding with application to speech spectrum quantization. *IEEE Trans. on Audio, Speech and Language Proc.*, 14(2):524–532, mar 2006.

[113] A. D. Subramaniam and B. D. Rao. Pdf optimized parametric vector quantization of speech line spectral frequencies. *IEEE Trans. Speech and Audio Proc.*, 11(2):130–142, mar 2003.

[114] G. J. Sullivan. Efficient scalar quantization of exponential and laplacian random variables. *IEEE Trans. on Information Theory*, 42(5):1365–1374, sept 1996.

[115] ITU-T T.82. *Information technology - coded representation of picture and audio information - progressive bi-level image compression.* 1993.

[116] L. Tancerel, S. Ragot, V. T. Ruoppila, and R. Lefebvre. Combined speech and audio coding by discrimination. *IEEE Workshop on Speech Coding,*, pages 154–156, Sept 200.

[117] M. Tasto and P. A. Wintz. Image coding by adaptive block quantization. *IEEE Trans. on Comm. Tech.*, 19(6):957–972, dec 1971.

[118] D. Taubman, E. Ordentlich, M. Weinberger, G. Serouss, I. Ueno, and F. Ono. Embedded block coding in jpeg2000. *Proc. IEEE International Conference on Image Processing (ICIP)*, 2:33–36, sept 2000.

[119] D. S. Taubman and M. W. Marcellin. *JPEG2000: Image Compression Fundamentals, Standards and Practice.* Springer, 2001.

[120] D. S. Taubman and M. W. Marcellin. Jpeg2000: standard for interactive imaging. *Proc. IEEE*, 90(8):1336–1357, aug 2002.

[121] M. J. Tsai, J. D. Villasenor, and F. Chen. Stack-run coding for low bit rate image communication. *in Proc. ICIP.*

[122] M. J. Tsai, J. D. Villasenor, and F. Chen. Stack-Run image coding. *IEEE Trans. Circuits Syst. Video Technol.*, 6:519–521, oct 1996.

[123] P. P. Vaidyanathan. Quadrature mirror filter, m-band extensions and perfect-reconstruction techniques. *IEEE ASSP Magazine*, pages 4–20, 1987.

[124] I. Varga. On new audio codec specification. *IEEE 8th Workshop on Multimedia Signal Proc.*, pages 128–131, oct 2006.

[125] J. Villasenor, M. J. Tasai, and F. Chen. Method for reducing storage requirements for digital data.

[126] I. H. Witten, R. M. Neal, and J. G. Cleary. Arithmetic coding for data compression. *Communications of the ACM*, 30:520–540, June 1987.

[127] M. Wolters, K. Kjörling, and H. Purnhagen. A closer look into MPEG-4 High Efficiency AAC. *Presented at the AES 115th Convention*, page Preprint 5871, Oct 2003.

[128] M. Xie, D. Lindbergh, and P. Chu. From ITU-T G.722.1 to ITU-T G.722.1 annex c: anew low-complexity 14 khz bandwidth audio coding standard. *Journal of Multimedia*, 2(2):65–76, April 2007.

[129] D. Y. Zhao. *Model based speech enhancement and coding.* PhD thesis, School of Electrical Engineering KTH, Stockholm, 2007.