UNIVERSITÉ DE SHERBROOKE
Faculté de génie
Département de génie électrique et génie informatique

NOUVELLES TECHNIQUES DE QUANTIFICATION VECTORIELLE ALGÉBRIQUE BASÉES SUR LE CODAGE DE VORONOI – APPLICATION AU CODAGE AMR-WB+

Thèse de doctorat (Ph.D.)

Spécialité: génie électrique

Stéphane RAGOT

Sherbrooke (Québec), Canada Mai 2003

RÉSUMÉ

L'objet de cette thèse est l'étude de la quantification (vectorielle) par réseau de points et de son application au modèle de codage audio ACELP/TCX multi-mode. Le modèle ACELP/TCX constitue une solution possible au problème du codage audio universel – par codage universel, on entend la représentation unifiée de bonne qualité des signaux de parole et de musique à différents débits et fréquences d'échantillonnage. On considère ici comme applications la quantification des coefficients de prédiction linéaire et surtout le codage par transformée au sein du modèle TCX ; l'application au codage TCX a un fort intérêt pratique, car le modèle TCX conditionne en grande partie le caractère universel du codage ACELP/TCX.

La quantification par réseau de points est une technique de quantification par contrainte, exploitant la structure linéaire des réseaux réguliers. Elle a toujours été considérée, par rapport à la quantification vectorielle non structurée, comme une technique prometteuse du fait de sa complexité réduite (en stockage et quantité de calculs). On montre ici qu'elle possède d'autres avantages importants : elle rend possible la construction de codes efficaces en dimension relativement élevée et à débit arbitrairement élevé, adaptés au codage multi-débit (par transformée ou autre) ; en outre, elle permet de ramener la distorsion à la seule erreur granulaire au prix d'un codage à débit variable.

Plusieurs techniques de quantification par réseau de points sont présentées dans cette thèse. Elles sont toutes élaborées à partir du codage de Voronoï. Le codage de Voronoï quasi-ellipsoïdal est adapté au codage d'une source gaussienne vectorielle dans le contexte du codage paramétrique de coefficients de prédiction linéaire à l'aide d'un modèle de mélange gaussien. La quantification

vectorielle multi-débit par extension de Voronoï ou par codage de Voronoï à troncature adaptative est adaptée au codage audio par transformée multi-débit.

L'application de la quantification vectorielle multi-débit au codage TCX est plus particulièrement étudiée. Une nouvelle technique de codage algébrique de la cible TCX est ainsi conçue à partir du principe d'allocation des bits par remplissage inverse des eaux.

REMERCIEMENTS

J'aimerais tout d'abord remercier mon directeur de recherche, Roch Lefebvre, pour m'avoir offert d'excellentes conditions de doctorat et m'avoir permis de développer mes intérêts personnels de recherche, tout en conciliant les intérêts de la compagnie VoiceAge. La confiance qu'il a su m'accorder a largement contribué à l'accomplissement de ce travail de doctorat. Je voudrais également exprimer ma gratitude aux personnes qui ont appuyé mes travaux au sein de VoiceAge – dans le cadre du projet AMR-WB+ –, et plus particulièrement Redwan Salami, Vice-Président R&D.

Je remercie sincèrement Robert M. Gray, professeur à l'Université Stanford, Claude Lamblin, ingénieur-chercheur à France Télécom R&D, ainsi que Bernard Colin, professeur de mathématiques à l'Université de Sherbrooke, pour m'avoir fait l'honneur d'évaluer ce travail et fait bénéficier de leur expertise. Je suis tout particulièrement reconnaissant envers Claude Lamblin pour sa lecture méticuleuse du manuscrit de la thèse et pour ses nombreuses suggestions. Ma reconnaissance va également à Bernard Colin, pour avoir accepté d'être rapporteur de cette thèse et pour ses corrections détaillées.

J'adresse un grand merci à tous les membres du groupe de recherche en codage – à l'Université de Sherbrooke comme à VoiceAge – qui m'ont soutenu et encouragé, et avec qui j'ai eu le plaisir de travailler. Ces remerciements sont en particulier destinés à mes collaborateurs directs au sein du projet AMR-WB+ : Bruno Bessette, Vesa T. Ruoppila – et plus récemment "Joe" Thiemann. J'aimerais aussi remercier Danielle Poirier, secrétaire du groupe, pour avoir réglé de façon professionnelle et efficace tous les détails administratifs avant ma soutenance, ainsi que pour sa bonne humeur et sa

sympathie. Je remercie également Philippe Gournay, ingénieur-chercheur dans le groupe de codage, pour m'avoir rendu service à plusieurs reprises.

Un merci chaleureux aux étudiants, stagiaires du groupe de codage et du département de génie électrique que j'ai cotoyé : Ludovic, Antonin, Francis, Hassan, Ali, Maxime, Mikko, Emmanuelle, Jérôme, Sidi, Luis et Ramin. Les à-cotés de la recherche que nous avons partagé sont autant de bons moments passés à Sherbrooke. Je tiens à remercier également, Jean-Luc, Guillaume et Rémy, du groupe de codage, pour avoir relu certaines parties du manuscrit. En outre, les échanges de point de vue que j'ai pu avoir avec Rémy sur la recherche ont été stimulants.

Je ne pourrais pas clore ces remerciements sans exprimer ma reconnaissance profonde à Jean-Pierre Adoul, qui m'a introduit au codage algébrique en 1996-97 à Sherbrooke et grâce à qui je suis resté dans le groupe de codage. Cette thèse est en grande partie tributaire de son enseignement initial, de sa confiance et de ses encouragements reçus à mes débuts à Sherbrooke.

Je tiens aussi à remercier Svetlana. Enfin, je remercie de tout cœur ma famille – en particulier mes parents et mon frère – pour leur soutien indéflectible malgré les longues années passées de l'autre côté de l'Atlantique.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

LISTE DES ABRÉVIATIONS ET DES SYMBOLES IMPORTANTS

$\mathbb{N}$	ensemble des entiers naturels
$\mathbb{N}^*$	ensemble des entiers strictement positifs
$\mathbb{N}\setminus\{0,1\}$	ensemble des entiers supérieurs ou égaux à 2
$\mathbb{R}$	ensemble des nombres réels
$\mathbb{C}$	ensemble des nombres complexes
M^t	transposée de la matrice M
$\lfloor . \rfloor$	arrondi à l'entier inférieur
$\lceil . \rceil$	arrondi à l'entier supérieur
$[.]$	arrondi à l'entier le plus proche
$[(x_1, \cdots, x_N)]$	arrondi des composantes d'un vecteur à l'entier le plus proche
mod	modulo scalaire
N	dimension (taille de vecteurs)
K	nombre de sous-vecteurs (de taille identique)
Λ	réseau de points quelconque
$M(\Lambda)$	matrice génératrice d'un réseau de points
$G(\Lambda)$	moment normalisé d'ordre 2 (constante de quantification) d'un réseau de points
$V(\Lambda)$	région de Voronoï de Λ (pour la norme euclidienne) associée à l'origine
$\gamma_g(\Lambda)$	gain granulaire du réseau Λ
r	ordre de l'extension de Voronoï
$C^{(r)}$	extension de Voronoï de C d'ordre r
n	numéro de dictionnaire
n^E	code de longueur variable représentant un numéro de dictionnaire
i	indice d'un mot de code
$\mathbf{k}$	indice de Voronoï (vectoriel)
Λ/Λ'	partitions du réseau Λ induite par le sous-réseau Λ'
$\lvert\Lambda/\Lambda'\rvert$	ordre (taille) de la partition Λ/Λ'
$[\Lambda/\Lambda']$	représentants des cosets de Λ' dans Λ

3GPP	3rd Generation Project Partnership
AAC	Advanced Audio Coding (partie du standard MPEG-4 Audio)
AAC+ (aacPlus)	Version de AAC avec extension de bande
ACELP	Algebraic Code-Excited Linear Prediction
ATCELP	Adaptive Transform and CELP
AMR	Adaptive Multi-Rate
AMR-WB	Adaptive Multi-Rate WideBand (spécification du 3GPP pour le codage de parole en bande élargie comprenant 9 modes de 6.6 à 23.85 kbit/s)
AMR-WB+	Extension du codage AMR-WB pour la musique et la stéreo
CDMA	Code Division Multiple Access
CELP	Code-Excited Linear Prediction
CNG	Confort Noise Generator
DMT	Discrete Multi Tone
DTX	Discontinuous Transmission
EFR	Enhanced Full Rate
FELP	Frequency-Excited Linear Prediction
FFT	Fast Fourier Transform
G.722.1	Recommandation de l'UIT-T de codage audio en bande élargie à 24, 32 kbit/s
G.722.2	Recommandation de l'UIT-T équivalente à l'AMR-WB
G.729 (A/E)	Recommandation de l'UIT-T de codage de parole en bande étroite (annexes A/E)
ISF	Imittance Spectral Frequency
ISO	International Standardization Organization
ISPP	Interleaved Single-Pulse Permutations
LBG	Linde, Buzo, Gray
LPC	Linear Predictive Coding
LVQ	Lattice Vector Quantization
LSF	Line Spectral Frequency
MA	Moving Average
MLT	Modulated Lapped Transform
MMS	Multimedia Messaging Service
MPEG	Moving Picture Experts Group
MTPC	Multimode Transform Predictive Coder
OFDM	Orthogonal Frequency-Division Multiplexing
OTC	Orthogonal Transform Coding
PCM	Pulse Coded Modulation
PLVQ	Pyramid Lattice Vector Quantization
PVQ	Pyramid Vector Quantization
PSS	Packet Switched Services
PSM	Packet Switched Multimedia
RELP	Residual-Excited Linear Prediction
RSB	Rapport signal-sur-bruit
SMV	Selectable Mode Vocoders
SLVQ	Spherical Lattice Vector Quantization

SVQ	Scalar-Vector Quantizer
TCQ	Treillis Coded Quantization
TCVQ	Treillis Coded Vector Quantization
TCX	Transform Coded eXcitation
TNS	Temporal Noise Shaping
TPC	Transform Predictive Coder
UIT-T	Union Internationale des Télécommunicaitons – secteur Télécommunications
VAD	Voice Activity Detector

Chapitre 1

INTRODUCTION

Cette thèse porte sur le codage audio, c'est-à-dire sur la compression (ou codage avec perte) des signaux audio-fréquences tels que la parole et la musique, à des fins de communications (transmission ou stockage) et de reproduction. Les méthodes associées constituent une technologie clé – on les qualifie d'*enabling technology* en anglais –, dans le sens où elles conditionnent la qualité des systèmes de télécommunications et multimédia (téléphonie cellulaire numérique, diffusion audio sur Internet, stockage audio de qualité CD, etc.).

Cette thèse s'inscrit dans un cadre de recherche lié au développement d'un modèle de codage, appelé *ACELP/TCX multi-mode*, introduit dans [1] et constituant une réponse possible au problème du codage universel. Le codage audio universel est un besoin clair de l'industrie et se définit de façon générale comme la recherche d'un modèle unique de *bonne qualité*, capable de traiter efficacement des signaux audio cadencés à *différentes fréquences d'échantillonnage* à *débits différents* et *indépendamment de leurs caractéristiques* intrinsèques (parole, musique, etc.) pour des applications à différents retards. Le défi principal consiste en fait à trouver une représentation unifiée de la parole et de la musique à bas débit. Bien que le contexte de la thèse soit du ressort du codage audio et de la modélisation de signaux, les travaux présentés ici ont été volontairement orientés vers des problèmes de quantification reliés au modèle ACELP/TCX.

Cette introduction a pour but de dresser un bref état de la recherche en codage audio, en insistant sur le problème du codage universel, et de donner un aperçu des travaux présentés dans la thèse. On y justifie en particulier la restriction des travaux à la quantification vectorielle algébrique. On donne à la section 1.1 un survol du domaine de recherche. La section 1.2 se concentre sur le problème du codage universel, et le modèle ACELP/TCX multi-mode. On y décrit aussi brièvement le modèle de codage appelé AMR-WB+ développé dans le cadre de la standardisation 3GPP. Un aperçu de la thèse est donné à la section 1.3, où l'on précise en particulier l'originalité et l'intérêt pratique des contributions des travaux présentés et l'organisation globale du document.

1.1 Domaine de recherche : codage audio

1.1.1 Chaîne de communication audio avec une source numérique

Le problème considéré ici se rapporte à la représentation numérique des signaux audio, à des fins de communications et de reproduction. La chaîne de communication associée est illustrée à la figure 1.1. À son entrée, on trouve la source audio, naturelle ou synthétique, et à sa sortie, un récepteur quelconque incluant l'oreille humaine. Le codage audio apparaît donc comme un intermédiaire de communication ; son rôle principal est de réduire le débit binaire de la source (numérique) afin d'utiliser les ressources du canal avec parcimonie.

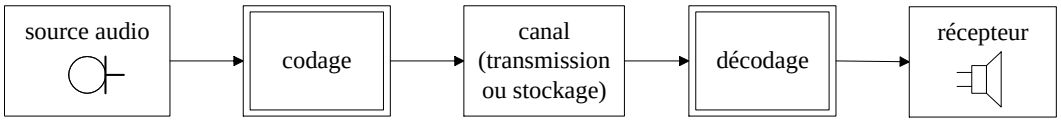


Figure 1.1: Chaîne de communication audio.

On rappelle au tableau 1.1 les caractéristiques usuelles des signaux audio. Le codage de parole a d'abord été considéré en bande étroite pour la téléphonie ; la bande élargie a été définie pour l'audio- et la visio-conférence [2, 3]. Le codage audio (incluant la musique) à plus large bande s'est développé à partir du disque compact (CD) et de la diffusion audio grand public. Le codage de

parole est plutôt orienté vers des applications de communications vocales bi-directionnelles, à retard et complexité faibles, tandis que le codage audio est avant tout destiné à des applications de diffusion et de stockage, dans lesquelles le retard et la complexité de codage importent moins.

TABLEAU 1.1: Fréquences d'échantillonnage usuelles et bandes utiles des signaux audio (monophoniques).

Fréquence d'échantillonnage (Hz)	Bande utile (Hz)	Résolution (bits)	Débit binaire (kbit/s)	Applications	Qualité
8000	300-3400	8	64	Téléphonie	Bande étroite, qualité téléphonique (*toll quality*)
11025 (1/4 CD)	10-5000	16	176.4	PC multimédia	
16000	50-7000	16	256	Audio-/vidéo-conférence, radio sur Internet, téléphonie 3G	Bande élargie (*face-to-face*), radio AM
22050 (1/2 CD)	10-10000	16	352.8	PC multimédia	
32000	20-15000	16	512	TVHD, radio numérique (DAB), NICAM	Radio FM
44100 (CD)	10-20000	16	705.6 (×2)	CD audio	CD, Hi-fi
48000	> CD	24		DAT, studio	
96000	> CD			SACD, studio	

D'une manière générale, le codage audio consiste à éliminer les éléments constitutifs de la source qui sont redondants – au sens statistique – ou inaudibles (non pertinents au récepteur). Ce principe implique de disposer d'une connaissance précise à la fois de la source et du récepteur.

On classifie les signaux audio en silence, bruit, parole, musique et signaux mixtes (mélanges). Le signal de parole est étudié en détail par exemple dans [4]. Par souci de concision, on ne revoit pas ici les notions élémentaires que sont les phonèmes, le modèle source-filtre, et les caractéristiques intrinsèques de la parole (formants, impulsions glottales, pitch, radiation des lèvres, etc.). On trouve par exemple dans [5] des notions sur les signaux de musique.

L'oreille humaine, l'audition et la perception auditive, sont étudiées dans [6, 7]. Là encore, on passe ici sous silence les notions élémentaires de bandes critiques, masquages temporel et fréquentiel, seuil d'audition, courbe de masquage, etc.

1.1.2 Perspective historique du codage audio et approches de codage

Historiquement, le codage audio s'est développé autour de deux approches de représentation de la source sonore :

1. Le codage de forme d'onde, issu de la théorie du codage de source avec un critère de fidélité (ou théorie de la distorsion) de [8, 9, 10] et construit à partir du format PCM (*Pulse Code Modulation*) [11].

2. Le codage paramétrique de parole (ou vocodage) basé essentiellement sur une modélisation du signal – on fait généralement remonter cette approche jusqu'à Dudley [12].

La frontière entre codage de forme d'onde et vocodage s'est effacée avec le développement de techniques hybrides [13, 14], incluant le codage paramétrique basé sur le principe d'analyse par synthèse [15, 16, 17, 18, 19].

La source n'est cependant pas le seul élément dans la chaîne de communications audio ; on peut aussi chercher à exploiter les limites du récepteur : l'oreille humaine. La prise en compte de la perception humaine a ainsi conduit à des évolutions importantes du codage de forme d'onde, avec l'incorporation de filtres perceptuels [20, 21] puis de modèles de masquage [22, 23, 24] dans le critère de fidélité.

Dans cette thèse on s'intéresse avant tout aux techniques de codage à bas débit conduisant à une synthèse audio de bonne qualité, telles que le codage de parole de type CELP [19] et le codage audio perceptuel [24]. On travaille donc à des débits généralement dans la gamme 8–24 kbit/s en bande élargie. Les techniques de base du codage de forme d'onde ne sont pas revues – elles sont présentées en détail dans [25]. Le codage paramétrique de la parole (vocodage) n'est pas étudié ici car sa qualité est limitée par la modélisation imparfaite du signal. On peut toutefois en rappeler les techniques de base : le vocodeur à canaux [12], le synthétiseur à formants [26] ou le vocodeur à prédiction linéaire [27] (avec des variantes sur la représentation de l'enveloppe spectrale) ; les modèles plus évolués [28, 29, 30, 31, 32, 33] sont basés sur la prédiction linéaire. On peut aussi signaler à très bas débit

des approches de codage sémantique telles que le codage phonétique [34], le codage par indexation d'unités sonores [35] et la représentation audio structurée [36]. Les techniques connexes au codage audio, comme le pré-traitement (réduction de bruit), l'annulation d'écho, ou le codage multi-débit (sélection de modes, VAD, CNG, et DTX) ne sont pas revues ici.

Pour se fixer les idées, les ordres de grandeurs de débits (en kbit/s) communément utilisés en codage de parole sont rappelés au tableau 1.2.

TABLEAU 1.2: Gammes de débit en codage de parole en bande étroite et bande élargie.

Débit de codage (en kbit/s)	Bande étroite	Bande élargie
Très bas débit	< 2.4	< 4.8
Bas débit	2.4-4.8	4.8-9.6
Débit intermédiaire	4.8-16	9.6-32
Haut débit	≥ 16	≥ 32

L'évolution technologique en codage audio est reflétée par les nombreux standards adoptés dans les organismes de normalisation telles que l'UIT-T, l'ETSI, le TIA ou l'ISO/IEC. On trouve des synthèses sur ce sujet dans [37, 38, 39].

1.1.3 Axes de recherche en codage audio

Les axes représentatifs de la recherche actuelle en codage de parole et audio sont représentés à la figure 1.2, sans chercher à les classer de façon cohérente. Parmi ces axes de recherche, on retrouve les attributs (en partie conflictuels) du codage audio : qualité vs débit, complexité, retard, robustesse aux erreurs de canal. On comprend donc pourquoi certains axes de la figure 1.2 se recoupent.

Les problèmes de recherche enumérés ici dépendent souvent des applications et des services visés, ainsi que des signaux audio traités. Par exemple, le codage faible retard convient aux applications de communications bi-directionnelles temps-réel. Le codage faible complexité est primordial en téléphonie mobile, dans les noeuds de réseaux de télécommunications, etc. Le codage multi-mode, à débit variable (de type AMR ou SMV) convient à la téléphonie cellulaire. Le codage sémantique

de parole (à très bas débit) est plutôt destiné aux applications militaires. Le codage multi-canal, où le nombre de canaux correspond au nombre de haut-parleurs (par exemple, canaux gauche, droit, central, surrond, avant, arrière), convient aux applications comme le cinéma. L'audio spatialisé (ou ambisonie, restitution sonore 3D) tombe plus dans le domaine de l'acoustique que du codage proprement dit et rejoint le problème de codage multi-canal ; l'idée consiste à créer une illusion spatiale en entourant l'auditeur de haut-parleurs, qui servent à approximer (au niveau de l'auditeur) le champ sonore associé à des sources réelles.

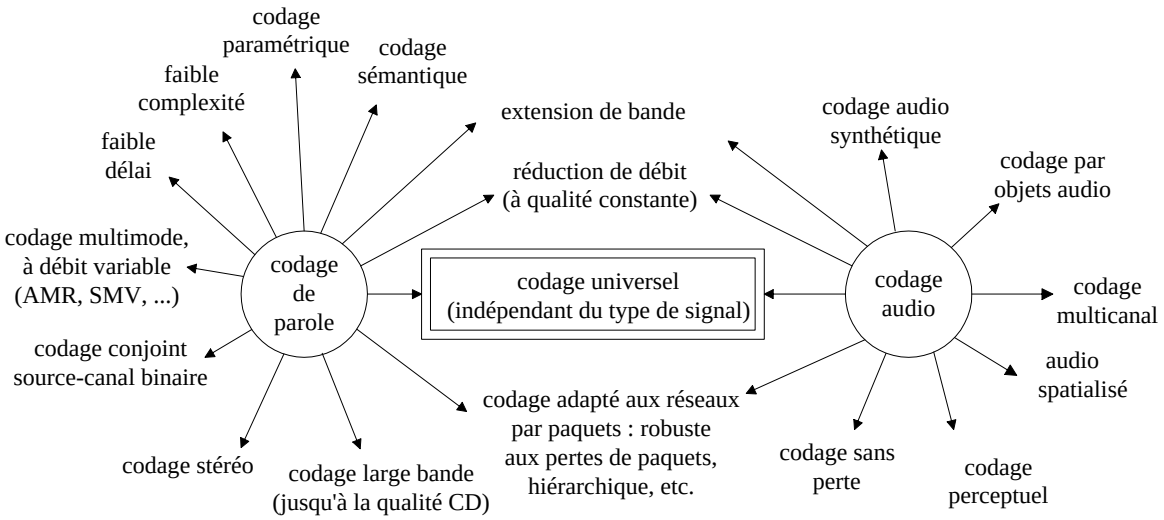


Figure 1.2: Axes de recherche importants en codage de parole et audio.

Les travaux de cette thèse s'inscrivent plus spécifiquement dans l'axe appelé ici *codage universel*. Néanmoins, les contributions présentées ici se rapportent indirectement aux problèmes de base du codage : la réduction de débit à faible complexité.

1.2 Contexte des travaux : codage universel des signaux audio et AMR-WB+

La recherche d'un format universel de codage audio à bas débit indépendant du type de signal d'entrée – que ce soit de la parole, de la musique, etc. – peut être considérée comme le "saint Graal" du codage audio. La théorie du codage universel s'est développée avant tout à partir d'une approche théorique statistique du codage de source. On dresse ici un panorama rapide de cette théorie et des techniques générales associées. Néanmoins, traditionnellement, le codage audio à bas débit repose plutôt sur une approche expérimentale de modélisation et de représentation de signaux. On adopte ici cette approche en définissant le problème du codage audio universel avant tout comme un problème de représentation temps-fréquence, donc un problème de modélisation.

1.2.1 Théorie du codage universel de source (approche statistique)

Le concept de codage universel de source a été introduit par Kolmogorov en 1965 pour qualifier des algorithmes capables de compresser des données sans connaissance a priori de la distribution de la source [40]. Ce type d'incertitude sur la source peut être également considéré (par exemple) en prédiction et estimation [41]. Il s'agit d'un problème important, car on dispose rarement en pratique d'une connaissance statistique complète sur la source à coder. Différents types d'universalité sont définis rigoureusement dans [42] pour le codage sans perte et dans [43] pour le codage avec perte. Dans tous les cas, un codeur est dit universel si pour une large classe de sources il peut atteindre (au moins asymptotiquement) les performances optimales de codage – l'entropie de la source en codage sans perte ou la borne débit-distorsion en codage avec perte.

Le codage universel a d'abord été étudié dans sa version sans perte et de façon parcellaire avant d'être formalisé en 1973 par Davisson [42]. Ces développements ont ensuite été étendus au codage avec perte – ils sont revus en partie dans [44]. Parmi les algorithmes classiques de codage universel sans perte, on trouve le codage de Huffman adaptatif [45, 46, 47], le codage par intervalles de répétition de Ryabko [48, 49] (redécouvert dans [50, 51]), les algorithmes de Lempel-Ziv [52, 53],

le codage arithmétique adaptatif [54], le codage par transformation de Burrows-Wheeler [55] et le codage universel des entiers [56, 57, 58, 59, 60] incluant le codage multi-dictionnaires de Rice [61]. Ces algorithmes sont par nature adaptatifs.

On s'intéresse ici plutôt au codage avec perte. Il est connu qu'un quantificateur optimisé pour une source a des performances sous-optimales dès lors que la distribution réelle de la source ne correspond pas au modèle adopté pour l'optimisation (*source mismatch*) [62]. Ce problème englobe également les cas où la distribution est connue partiellement (moyenne inconnue, variance inconnue, etc.) [63]. La quantification peut être rendue plus robuste à de telles incertitudes en employant une technique de codage de type gain-forme et/ou à moyenne supprimée [64]. Une approche alternative et plus générale consiste à employer une technique de quantification universelle. La recherche sur le codage de source avec perte tombe dans deux catégories : la preuve de l'existence et la construction de codes d'une part, et les algorithmes réalisables et efficaces d'autre part. L'existence de codes a été montrée dans [43, 65, 66, 67, 68, 69, 70]. Des algorithmes sont décrits dans [71, 72, 73, 74, 75, 76, 77, 78, 79]. En particulier dans [71], la quantification algébrique par réseau de points est suivie par un codage entropique sans perte et universel. De plus, la quantification par adaptation de dictionnaires est présentée dans [64].

On trouve également dans ce cadre statistique quelques développements en codage universel des signaux réels, comme le codage d'images par transformée avec classification [80] ou transformation adaptative [81].

1.2.2 Codage audio universel : problématique, solutions actuelles et cadre de standardisation

Les technologies définissant l'état de l'art en codage audio de bonne qualité à un débit inférieur à 2 bit par échantillon peuvent être classées en deux groupes [82] : le codage de parole (temporel) et le codage de musique (fréquentiel). De ce point de vue, le codage universel apparaît comme un problème de modélisation temps-fréquences.

Le codage prédictif de parole convient difficilement au codage universel à bas débit. En effet, il est fondé sur un modèle de production de la parole, inapplicable à l'audio en général. À l'opposé, le codage audio perceptuel est fortement limité à bas débit pour les signaux de parole. Le codage fréquentiel utilisé ne prend pas en compte explicitement les structures importantes de la parole (variation temporelle fine du pitch, formants) et n'est pas adapté à la représentation des signaux non-stationnaires et transitoires tels que les attaques et les plosives. Ces limites de représentation peuvent être cependant repoussées en employant par exemple un fenêtrage adaptatif [83, 84], une commutation de bancs de filtres [85, 86], une réduction de pré-écho par post-filtrage adaptatif [87], etc.

Plusieurs techniques ont été développées pour répondre au problème du codage universel :

- Codage prédictif par transformée : TCX [88, 89, 90], TPC [91, 92], OTC [93] ;
- Codage sinusoïdal [94, 95, 96, 97, 98, 99] ;
- Codage multi-mode : ACELP/TCX [1] avec classification en boucle fermée, ACELP/G.722.1 [100], ATCELP [101], codage avec prédiction linéaire mixte [102, 103] et MPTC [82] avec classification en boucle ouverte ;
- Codage perceptuel par transformée ou en sous-bandes, avec mise en forme temporelle du bruit de codage (TNS) [104, 105], pré/post-filtrage [106] ou quantification spécifique [107, 108] ;
- Codage hiérarchique, par exemple par encapsulation CELP+transformée [109] ou directement par transformée [110].

Toutes ces techniques diffèrent en qualité, complexité, retard, etc.

Le codage audio universel à bas débit est encore un problème ouvert. Historiquement, la recherche sur ce sujet a été stimulée par des activités de standardisation UIT-T et ISO/MPEG, ainsi que par les besoins de l'industrie (diffusion radio par Internet, audio- et vidéo-conférence, stockage multimédia, etc.). La question Q.20/16 de l'UIT-T (SG16 WP3/16), visant à recommander une

technique de codage en bande élargie de bonne qualité pour la parole et la musique à 16, 24 et 32 kbit/s [111], a souligné la difficulté du codage audio universel à bas débit. Elle a abouti à la recommandation G.722.1 opérant à 24 et 32 kbit/s dans des conditions restrictives. C'est dans ce cadre de standardisation qu'on été développées les techniques de codage de type TCX, TPC, ACELP/TCX, ATCELP, MPTC. En parallèle, les activités MPEG-4 Audio [112, 113] ont motivé la recherche en codage sinusoïdal et codage hiérarchique, en plus d'avoir produit le standard AAC qui intègre des améliorations du codage par transformée. Plus récemment, la problématique du codage audio universel a été relancée dans l'activité 3GPP du groupe TSG-SA4 PSM visant à sélectionner une technique de codage audio pour des services de diffusion (PSS) et de messagerie multimédia (MMS) [114]. Cette standardisation est en cours et oppose en particulier les modèles de codage AMR-WB+ et AAC+, qui sont respectivement des versions améliorées du codage ACELP/TCX et AAC. Par ailleurs, le codage audio hiérarchique fait partie des problèmes posés dans la nouvelle question Q.9/16 de l'UIT-T (SG16).

1.2.3 Modèle ACELP/TCX multi-mode et codage AMR-WB+

Les travaux de cette thèse s'appuient sur un modèle spécifique de codage – constituant une réponse possible au problème du codage audio universel : le modèle ACELP/TCX [1].

Principe du modèle ACELP/TCX multi-mode

Le principe du codage ACELP/TCX est schématisé à la figure 1.3. Il s'agit d'un codage multi-mode avec décision en boucle fermée, alternant entre codage temporel (ACELP) et codage fréquentiel (TCX). Sa stratégie s'apparente aux approches de codage multi-dictionnaire, tel que le codage de Rice en codage universel sans perte des entiers [61], et surtout à la quantification vectorielle par "filet protecteur" (*safety-net vector quantization* en anglais) de [115].

Le principe du modèle ACELP/TCX est dû à Adoul.

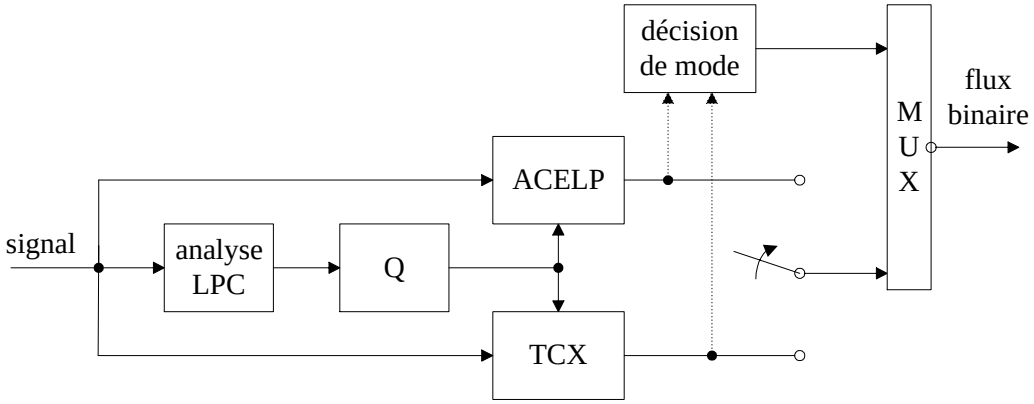


Figure 1.3: Vue simplifiée du codage ACELP/TCX.

Codage AMR-WB+

Le modèle ACELP/TCX multi-mode constitue le cœur du codage AMR-WB+ défini dans [116], tel qu'illustré à la figure 1.4. Ce dernier est une extension du codage de parole AMR-WB – décrit dans [117] – pour la musique et la stéréo, conçue pour des applications de messagerie multimédia (MMS) et de diffusion multimédia par paquets (PSS). Le mode ACELP est similaire au modèle ACELP du standard AMR-WB afin de minimiser le nombre de changement par rapport au standard AMR-WB et pour pouvoir rester inter-opérable avec le codage AMR-WB.

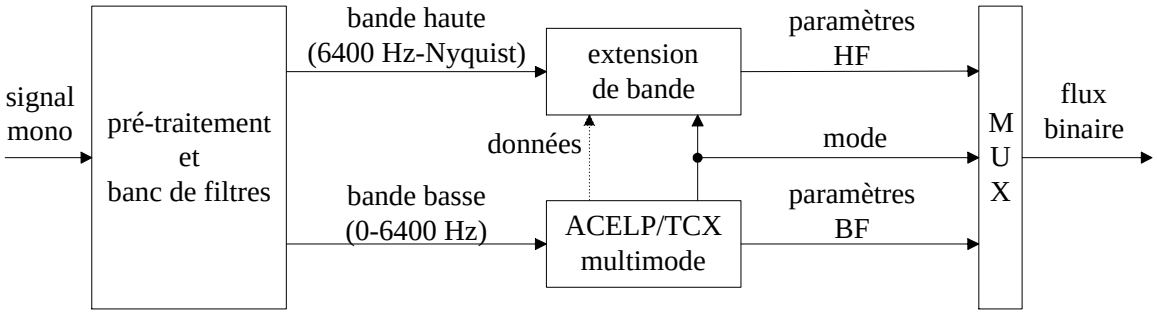


Figure 1.4: Vue simplifiée du codage AMR-WB+ mono.

Intérêt de la quantification vectorielle par réseau de points

La quantification par réseau de points est une technique de quantification uniforme exploitant la structure régulière des réseaux de points [118]. Celle-ci possède deux applications potentielles au sein du modèle ACELP/TCX multi-mode :

- Le codage TCX algébrique, tel qu'introduit dans [119] ;
- La quantification des coefficients de prédiction linéaire, explorée dans [120, 121, 122, 123, 124, 125, 126].

Le codage TCX algébrique tel que présenté dans [119] n'a été conçu que pour une application limitée : le codage de parole en bande élargie par trames de 20 ms à un débit de 16 kbit/s. Il a été étendu dans [1] au codage ACELP/TCX multi-mode. La quantification par réseau de points possède en effet de nombreux avantages dans ce contexte : elle est en particulier de complexité algorithmique faible, avec un stockage réduit, elle est surtout très bien adaptée au codage multi-débit.

La quantification vectorielle algébrique étant employée dans le mode TCX, son application à la quantification des coefficients de prédiction ne requiert qu'une complexité additionnelle limitée.

1.3 Aperçu de la thèse

1.3.1 Originalité et intérêt pratique des contributions

Cette thèse est orientée vers une approche algorithmique du codage de source et le développement de codes réalisables. Elle s'est déroulée dans le contexte de la recherche d'un modèle de codage audio universel et plus particulièrement dans le cadre du développement du modèle ACELP/TCX multi-mode. Elle ne reflète néanmoins qu'une partie des travaux réalisés dans ce cadre de recherche.

Pour assurer une cohérence globale à cette thèse, celle-ci a été articulée volontairement autour de la quantification vectorielle par réseau de points applicable au codage ACELP/TCX multi-mode.

La technique, appelée codage de Voronoï, constitue l'épine dorsale de la thèse. On présente une revue de la quantification vectorielle par réseau de points (chapitre 2) et des contributions originales détaillées ci-dessous :

- Le codage de Voronoï quasi-ellipsoïdal (chapitre 3), introduit partiellement dans [127, chap. 1] sous forme algorithmique (sans définition explicite), est complété ici en apportant plusieurs nouveautés :
  - Les codes de [127, chap. 1] sont définis comme des codes de Voronoï et les contraintes associées à leur définition sont clarifiées et démontrées.
  - Un algorithme général d'indexation est introduit.
  - Des algorithmes de saturation (par projection et réduction avec dichotomie) sont développés. La projection sur région de Voronoï repose sur une une recherche du vecteur pertinent.
  - Le problème de l'optimisation pour la source est étudié et un algorithme d'allocation des bits (via le modulo de Voronoï) de type glouton est présenté.

  Ces développements sont motivés par la quantification vectorielle paramétrique de coefficients de prédiction linéaire, telle que décrite dans [128, 129].

- Deux techniques de quantification vectorielle algébrique multi-débit sont présentées : l'extension de Voronoï (chapitre 4) et le codage par troncature de Voronoï adaptative (chapitre 5). Elles étendent la quantification vectorielle algébrique imbriquée de [119] et sont développées pour le codage TCX.

  L'extension de Voronoï est une technique due à Adoul ; cependant, l'idée sous-jacente n'a jamais été présentée ni expliquée. On s'attarde donc ici à définir cette technique à l'aide de la théorie des codes à cosets de Forney [130] et à la caractériser dans le cadre simple de la quantification d'une source gaussienne sans mémoire. Les travaux de cette thèse ont permis de contribuer au développement du système de quantification vectorielle algébrique multi-débit décrit au chapitre 4.

Le codage par troncature de Voronoï adaptative constitue une contribution importante de cette thèse, et comprend quelques aspects novateurs :

  - L'indexation multi-débit (avec un incrément de débit de 1 bit par dimension) par recherche de vecteur pertinent.
  - L'utilisation d'un réseau de points tourné afin de définir une indexation avec un incrément de débit de 1/2 bit par dimension.
  - Un système de quantification multi-débit basé sur le réseau RE_8.

  Cette technique est comparée à l'extension de Voronoï dans le cas d'une source gaussienne sans mémoire.

- Le codage TCX algébrique de [119] est étendu au chapitre 6 en utilisant les techniques de quantification des chapitres 4 et 5. La contribution principale concerne ici l'optimisation du gain TCX (global) dans un domaine logarithmique (ainsi que le multiplexage en plusieurs paquets et la correction des pertes de trame).

Tout au long de cette thèse, la source gaussienne sans mémoire (i.i.d) est utilisée dans les exemples. Cette source a l'avantage d'être simple ; la limite de Shannon est connue de façon analytique dans ce cas restreint. Néanmoins, les sources sans mémoire sont rarement de bons modèles pour des processus tels que les coefficients LPC, et les gains de compression sont souvent plus élevés quand la source est avec mémoire. Ainsi, les formes d'onde tels que les signaux résiduels ont en général des échantillons qui suivent une distribution laplacienne et qui sont décorrélés (et non indépendants). Dans cette thèse, la source gaussienne sans mémoire sert avant tout à caractériser (dans un cadre rigoureux) les outils de quantification vectorielle algébrique développés.

Plusieurs travaux significatifs ne sont ainsi pas présentés, pour se concentrer sur un sujet bien défini et ne pas alourdir cette thèse. Ces travaux sont en fait passés sous silence parce qu'ils sont déjà présentés dans [131, 132, 133, 134] ou, dans d'autres cas, parce qu'ils ne constituent pas une

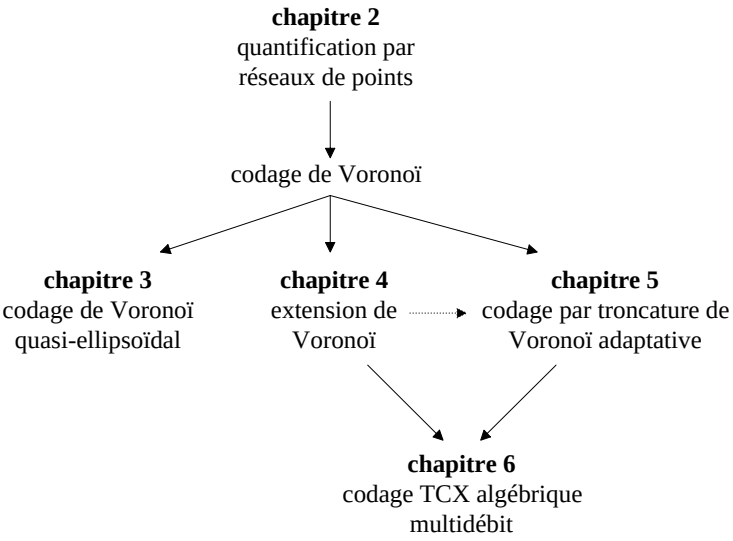
contribution très originale ou ne possèdent pas un intérêt pratique important par rapport à l'état de l'art. Ils comprennent :

- La quantification vectorielle des coefficients de prédiction linéaire par codes algébriques quasi-ellipsoïdaux [129] ou de type stochastique-algébrique – cette dernière a été introduite pour le codage en bande étroite dans [120, 121, 122] et adaptée au codage en bande élargie dans [125, 126, 135, 129].
- L'optimisation de l'indexation en quantification vectorielle prédictive des paires de raies spectrales (LSF) [136].
- Le codage MLT avec quantification vectorielle algébrique multi-débit [133].
- La réduction de complexité de la quantification vectorielle algébrique imbriquée [132].
- Le codage TCX en boucle ouverte par prédiction linéaire sur une échelle fréquentielle non-linéaire (en anglais *warped LPC*) [137], à partir d'une quantification vectorielle algébrique de dimension 4, 8 ou 16.
- Le codage multi-mode CELP/MLT [100, 131, 138].
- L'indexation de dictionnaires construits par réseaux de points robuste aux erreurs binaires et aux pertes de paquets [134].
- Le formatage du flux binaire en plusieurs paquets [139] et la correction de pertes de trames en codage AMR-WB+.

1.3.2 Organisation du document

Ce document est délibérément structuré comme une thèse à articles, de sorte que les chapitres puissent être lus de façon relativement indépendante les uns des autres. Chaque chapitre comprend ainsi sa propre bibliographie et ses propres annexes. Cette structure implique néanmoins certaines redondances, tant au niveau du texte – le codage de Voronoï et le modèle TCX sont par exemple

définis à plusieurs reprises – que de la bibliographie. Les liens logiques entre chapitres sont explicités ci-dessous :



Le chapitre 2 donne une revue de la quantification par réseau de points (en tant qu'outil, en dehors de son contexte applicatif). Les chapitres suivants prennent pour acquis certaines bases en quantification vectorielle par réseau. Le chapitre 3 est consacré au codage de Voronoï quasi-ellipsoïdal. Les chapitres 4 et 5 décrivent deux techniques alternatives de quantification vectorielle algébrique multi-débit : la quantification par extension de Voronoï et le codage par troncature de Voronoï adaptative, respectivement. Ces deux techniques sont appliquées dans le chapitre 6, dans lequel le codage TCX algébrique est étendu. Cette thèse se termine au chapitre 7 sur des conclusions générales.

BIBLIOGRAPHIE

[1] B. Bessette, R. Salami, C. Laflamme, and R. Lefebvre, "A wideband speech and audio codec at 16/24/32 kbit/s using hybrid ACELP/TCX techniques," in *Proc. IEEE Workshop on Speech Coding*, June 1999, pp. 7–9.

[2] X. Maitre, "7 kHz Audio Coding within 64 kbit/s," *IEEE J. Select. Areas Commun.*, vol. 6, pp. 283–288, Fev. 1988.

[3] P. Mermelstein, "G.722 – A new CCITT coding standard for digital transmission of wideband audio signals," *IEEE Comm. Mag.*, pp. 8–15, Jan. 1988.

[4] Calliope, *La parole et son traitement automatique*, Masson, Collection Technique et Scientifique des Télécommunications, 1989.

[5] N.H. Fletcher and T.D. Rossing, *The Physics of Musical Instruments*, Springer-Verlag, 1991.

[6] E. Zwicker and R. Feldkeller, *Psychoacoustique. L'oreille, récepteur d'information*, Masson, Collection Technique et Scientifique des Télécommunications, 1981.

[7] W.M. Hartmann, *Signals, Sound, and Sensation (Modern Acoustics and Signal Processing)*, Springer-Verlag, 1997.

[8] C.E. Shannon, "Coding theorems for a discrete source with a fidelity criterion," *IRE Nat. Conv. Rec.*, vol. 4, pp. 142–163, 1959.

[9] T. Berger, *Rate Distortion Theory*, Prentice-Hall, 1971.

[10] L.D. Davisson, "Rate-Distortion Theory and Applications," *Proc. IEEE*, vol. 60, pp. 800–808, Jul. 1972.

[11] B.M. Oliver, J.R. Pierce, and C.E. Shannon, "The Philosophy of PCM," *Proc. IRE*, vol. 36, pp. 1324–1331, Nov. 1948.

[12] H. Dudley, "The vocoder," *Bell Lab. Rec.*, vol. 18, pp. 122–126, Dec. 1939.

[13] B.S. Atal et al., "Voice-Excited Predictive Coding System for Low Bit Rate Transmission of Speech," in *Proc. ICC*, 1975, pp. 3037–3040.

[14] C.K. Un and D.T. Magill, "The Residual Excited Linear Prediction Vocoder with Transmission Rate Below 9.6 kbit/s," *IEEE Trans. Commun.*, vol. 23, Dec. 1978.

[15] L. Stewart, *Trellis data compression*, Thèse de doctorat, Stanford PhD dissertation, 1981.

[16] L. Stewart et al., "The design of trellis waveform coders," *IEEE Trans. Commun.*, vol. 30, pp. 702–710, Apr. 1982.

[17] B.S. Atal and J.R. Remde, "A new model of LPC excitation for producing natural sounding speech at low bit rates," in *Proc. ICASSP*, 1982, pp. 614–617.

[18] P. Kroon and E.F. Deprettere, "Experimental evalutation of different approaches to the multipulse coder," in *Proc. ICASSP*, 1984, pp. 10.4.1–10.4.4.

[19] M.R. Shroeder and B.S. Atal, "Code-excited linear prediction (CELP): High-quality speech at very low bit rates," in *Proc. ICASSP*, 1985, pp. 937–940.

[20] B.S. Atal and M.R. Schroeder, "Predictive coding of speech signals and subjective error criteria," *IEEE Trans. ASSP*, vol. 27, no. 3, pp. 247–254, 1979.

[21] B.S. Atal and M.R. Schroeder, "Optimizing predictive coders for minimum audible noise," in *Proc. ICASSP*, 1979, pp. 453–455.

[22] G. Theile, M. Link, and G. Stoll, "Low bit rate coding of high-quality audio signals," *AES Preprint*, , no. 2431, 1987.

[23] K. Brandenburg, "OCF – A new coding algorithm for high quality sound signals," in *Proc. ICASSP*, 1987, vol. 1, pp. 141–144.

[24] J.D. Johnston, "Transform coding of audio signals using perceptual noise criteria," *IEEE Trans. on Selected Areas in Commun.*, vol. 6, no. 2, pp. 314–323, 1988.

[25] N.S. Jayant and P. Noll, *Digital Coding of Waveforms – Principles and Applications to Speech and Video*, Prentice-Hall, 1984.

[26] D.H. Klatt, "Software for a cascade/parallel formant synthesizer," *J. Acoust. Soc. Am.*, vol. 67, pp. 971–995, 1980.

[27] B.S. Atal and S.L. Hanauer, "Speech Analysis and Synthesis by Linear Prediction of the Speech Wave," *J. Acoust. Soc. Am.*, vol. 50, pp. 637–655, 1971.

[28] J. Makhoul et al., "A mixed source model for speech compression and synthesis," *J. Acoust. Soc. Am.*, vol. 64, no. 6, pp. 1577–1581, 1978.

[29] D.W. Griffin and S. Lim, "Multi-Band Excitation Vocoder," *IEEE Trans. ASSP*, vol. 36, pp. 1223–1235, 1988.

[30] R.J. McAulay and T.F. Quatieri, *Sinusoidal coding*, chapter 4, pp. 121–173, Elsevier (W.B. Kleijn and K.K. Paliwal, eds.), 1995.

[31] W.B. Kleijn and J. Haagen, *Waveform Interpolation for Coding and Synthesis*, chapter 5, pp. 175–207, Elsevier (W.B. Kleijn and K.K. Paliwal, eds.), 1995.

[32] A.V. McCree and T.P. Barnwell III, "A mixed excitation LPC vocoder model for low bit rate specch coding," *IEEE Trans. ASSP*, vol. 3, no. 6, pp. 242–250, 1995.

[33] C. Laflamme, R. Salami, R. Matmti, and J.-P. Adoul, "Harmonic-stochastic eXcitation (hsx) speech coding below 4 kbit/s," in *Proc. ICASSP*, 1996, pp. 204–207.

[34] C.M. Ribeiro and I.M. Trancoso, "Phonetic vocoding with speaker adaptation," in *Proc. Eurospeech*, 1997.

[35] J. Cernosky, *Speech Processing using Automatically Derived Segmental Units : Application to Very Low Bit Rate Coding and Speaker Verification*, Thèse de doctorat, ESIEE / ENST Paris / VUT Brno / Université d'Orsay, Dec. 1998.

[36] E. D. Scheirer and B. L. Vercoe, "Saol: The mpeg-4 structured audio orchestra language," *Computer Music Journal*, vol. 23, no. 2, Summer 1999.

[37] R.V. Cox, *Speech Coding Standards*, chapter 2, pp. 45–78, Elsevier (W.B. Kleijn and K.K. Paliwal, eds.), 1995.

[38] V.K. Madisetti and D.B. Williams, *Digital Signal Processing (DSP) Handbook*, CRC Press/IEEE Press, 1997.

[39] A. Le Guyader, P. Philippe, and J.B. Rault, "Synthèse des normes de codage de la parole et du son (UIT-T, ETSI et ISO/MPEG)," *Ann. Télécommun.*, vol. 55, no. 9–10, pp. 425–441, Sept.–Oct. 2000.

[40] A.N. Kolmogorov, "Three approaches to the quantitative definition of information," *Probl. Inform. Transm.*, vol. 1, pp. 1–7, 1965.

[41] J. Rissanen, "Universal Coding, Information, Prediction, and Estimation," *IEEE Trans. Inform. Theory*, vol. 30, no. 4, pp. 629–636, Jul. 1984.

[42] L.D. Davisson, "Universal lossless coding," *IEEE Trans. Inform. Theory*, vol. 19, pp. 783–795, Nov. 1973.

[43] D.L. Neuhoff, R.M. Gray, and L.D. Davisson, "Fixed rate universal block source coding with a fidelity criterion," *IEEE Trans. Inform. Theory*, vol. 21, pp. 511–523, Sept. 1975.

[44] J.C. Kieffer, "A survey of the theory of source coding with a fidelity criterion," *IEEE Trans. Inform. Theory*, vol. 39, pp. 1473–1490, 1993.

[45] R. Gallager, "Variation on a theme by Huffman," *IEEE Trans. Inform. Theory*, vol. 24, pp. 668–674, 1978.

[46] D.E. Knuth, "Dynamic Huffman coding," *J. Algorithms*, pp. 163–180, 1985.

[47] J.S. Vitter, "Dynamic Huffman coding," *ACM Trans. Math. Software*, vol. 15, pp. 158–167, 1989.

[48] B. Ryabko, "Data compression by means of a book stack," *Probl. Inform.*, vol. 16, pp. 265–269, 1980.

[49] B. Ryabko, "A fast on-line adaptive code," *IEEE Trans. Inform. Theory*, vol. 38, pp. 1400–1404, Jul. 1992.

[50] P. Elias, "Interval and recency rank source coding: Two on-line adaptive variable-length schemes," *IEEE Trans. Inform. Theory*, vol. 33, no. 1, pp. 3–10, Jan. 1987.

[51] F.M.J. Willens, "Universal data compression and repetition times," *IEEE Trans. Inform. Theory*, vol. 35, no. 1, pp. 54–58, Jan. 1989.

[52] J. Ziv and A. Lempel, "A Universal Algorithm for Sequential Data Compression," *IEEE Trans. Inform. Theory*, vol. 23, no. 3, pp. 337–343, May 1977.

[53] J. Ziv and A. Lempel, "Compression of Individual Sequences via Variable-Rate Coding," *IEEE Trans. Inform. Theory*, vol. 24, no. 5, pp. 530–536, Sep. 1978.

[54] J. Rissanen, "A universal data compression system," *IEEE Trans. Inform. Theory*, vol. 29, pp. 656–663, Sep. 1983.

[55] M. Effros, K. Visweswariah, S.R. Kulkarni, and S. Verdú, "Universal Lossless Source Coding With the Burrows Wheeler Transform," *IEEE Trans. Inform. Theory*, vol. 48, no. 5, pp. 1061–1081, May 2002.

[56] P. Elias, "Universal codeword sets and representation of the integers," *IEEE Trans. Inform. Theory*, vol. 21, pp. 194–203, Mar. 1975.

[57] Q.F. Stout, "Improved prefix encodings of the natural numbers," *IEEE Trans. Inform. Theory*, vol. 26, pp. 607–609, Sep. 1980.

[58] P. Fenwick, "Punctured Elias Codes for variable-length coding of the integers," The University of Auckland, Dept. of Computer Science, Report No 137, Dec. 1996.

[59] R. Ahlswede, T.S. Han, and K. Kobayashi, "Universal coding of integers and unbounded search trees," *IEEE Trans. Inform. Theory*, vol. 43, pp. 669–682, Mar. 1997.

[60] D.P. Foster, R.A. Stine, and A.D. Wyner, "Universal codes for finite sequences of integers drawn from a monotone distribution," *IEEE Trans. Inform. Theory*, vol. 48, pp. 1713–1720, 2002.

[61] R.F. Rice and J.R. Plaunt, "Adaptive variable-length coding for efficient compression of spacecraft television data," *IEEE Trans. Commun.*, vol. 19, pp. 889–897, Dec. 1971.

[62] T.R. Fischer and R.M. Dicharry, "Vector Quantizer Design for Memoryless Gaussian, Gamma, and Laplacian Source," *IEEE Trans. Commun.*, vol. 32, no. 9, pp. 1065–1069, Sep. 1984.

[63] K. Sayood, *Introduction to Data Compression, 2nd ed.*, Morgan Kaufman, 2000.

[64] A. Gersho and R.M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic Publishers, 1992.

[65] J. Ziv, "Coding of sources with unknown statistics – Part II: Distortion relative to a fidelity criterion," *IEEE Trans. Inform. Theory*, vol. 18, pp. 389–394, May 1972.

[66] K.M. MacKenthun and M.B. Pursley, "Variable-rate universal block source coding subject to a fidelity criterion," *IEEE Trans. Inform. Theory*, vol. 24, pp. 340–360, May 1978.

[67] J.C. Kieffer, "A generalization of the Pushley-Davisson-Mackentun universal variable-rate coding thereom," *IEEE Trans. Inform. Theory*, vol. 23, no. 6, pp. 694–697, 1977.

[68] J.C. Kieffer, "A unified approach to weak universal source coding," *IEEE Trans. Inform. Theory*, vol. 24, no. 6, pp. 674–682, 1978.

[69] T. Linder, G. Lugosi, and K. Zeger, "Rates of convergence in the source coding theorem, in empirical quantizer design, and in universal lossy source coding," *IEEE Trans. Inform. Theory*, vol. 40, pp. 1728–1740, 1994.

[70] B. Yu and T.P. Speed, "A rate of convergence result for a universal d-semifaithful code," *IEEE Trans. Inform. Theory*, vol. 30, pp. 813–821, 1993.

[71] R. Zamir and M. Feder, "On universal quantization by randomized uniform/lattice quantizers," *IEEE Trans. Inform. Theory*, vol. 38, pp. 428–436, Mar. 1992.

[72] K. Zeger, A. Bist, and T. Linder, "Universal source coding with codebook transmission," *IEEE Trans. Commun.*, vol. 42, pp. 336–346, Feb.–Apr. 1994.

[73] T. Linder and K. Zeger, "Fixed Rate Universal Lossy Source Coding and Rates of Convergence for Memoryless Sources," *IEEE Trans. Inform. Theory*, vol. 41, no. 3, pp. 665–676, May 1995.

[74] E.H. Yang and J.C. Kieffer, "Simple universal lossy data compression schemes derived from the Lempel-Ziv algorithm," *IEEE Trans. Inform. Theory*, vol. 42, pp. 239–245, Jan. 1996.

[75] M. Effros, P.A. Chou, and R.M. Gray, "One-pass adaptive universal vector quantization," in *Proc. ICASSP*, 1994, vol. 5, pp. 625–628.

[76] A. Ortega and M. Vetterli, "Adaptive quantization without side information," in *Proc. ICIP*, 1994, vol. 3, pp. 856–860.

[77] Y. Steinberg and M. Gutman, "An algorithm for source coding subject to a fidelity criterion based on string matching," *IEEE Trans. Inform. Theory*, vol. 39, pp. 877–886, May 1993.

[78] M. Lightstone and S.K. Mitra, "Adaptive vector quantization for image coding in an entropy-constrained framework," in *Proc. ICIP*, 1994, vol. 1, pp. 618–622.

[79] A. Gersho and M. Yano, "Adaptive vector quantization by progressive codevector replacement," in *Proc. ICASSP*, 1985, pp. 133–136.

[80] M. Effros and P.A. Chou, "Weighted universal transform coding: Universal image compression with the karhunen-loève transform," in *Proc. ICIP*, 1995, vol. II, pp. 61–64.

[81] V.K. Goyal, J. Zhuang, and M. Vetterli, "Universal transform coding based on backward adaptation," in *Proc. IEEE Data Compression Conf.*, 1997.

[82] S. Ramprashad, "The multimode transform predictive coding paradigm," *IEEE Trans. on Speech and Audio Processing*, vol. 11, no. 2, pp. 117– 129, Mar. 2003.

[83] K. Brandenburg et al., "The ISO/MPEG Audio Codec: A Generic Standard for Coding of High Quality Digital Audio," *J. Audio Eng. Soc.*, vol. 42, Oct. 1994.

[84] M. Iwadare et al., "A 128 kb/s Hi-Fi Audio CODEC Based on Adaptive Transform Coding with Adaptive Block Size MDCT," *IEEE J. Select. Areas Commun.*, vol. 10, no. 1, Jan. 1992.

[85] D. Sinha and A.H. Tewfik, "Low bit rate transparent audio compression using adapted wavelets," *IEEE Transactions on Signal Processing*, vol. 41, no. 12, Dec. 1993.

[86] D. Sinha and J.D. Johnston, "Audio compression at low bit rates using a signal adaptive switched filterbank," *IEEE Transactions on Acoustics, Speech and Signal Processing*, 1996.

[87] Y. Mahieux and J.-P. Petit, "High-Quality Audio Transform Coding at 64 kbps," *IEEE Trans. Commun.*, vol. 42, no. 11, Nov. 1994.

[88] R. Lefebvre, R. Salami, C. Laflamme, and J.-P. Adoul, "High quality of wideband audio signals using transform coded excitation (TCX)," in *Proc. ICASSP*, 1994, vol. I, pp. 193–196.

[89] J.-P. Adoul and R. Lefebvre, *Wideband Speech Coding*, chapter 8, pp. 289–309, Elsevier (W.B. Kleijn and K.K. Paliwal, eds.), 1995.

[90] A. Jbira, N. Moreau, and P. Dymarski, "Low delay coding of wideband audio (20 Hz–15 kHz) at 64 kbps," in *Proc. ICASSP*, 1998.

[91] J.-H. Chen and D. Wang, "Transform Predictive Coding of Wideband Speech Signals," in *Proc. ICASSP*, 1996, pp. 275–278.

[92] J.-H. Chen, "A candidate for the ITU-T's new wideband speech coding standard," in *Proc. ICASSP*, 1997, vol. 2, pp. 1359–1362.

[93] N. Moreau and P. Dymarski, "Successive orthogonalizations in the multistage CELP coder," in *Proc. ICASSP*, 1992, pp. 61–64.

[94] M.M. Goodwin, *Adaptive signal models: Theory, Algorithms and Audio Applications*, Thèse de doctorat, University of California, Berkeley, Fall 1997.

[95] H. Purnhagen and M. Meine, "HILN – The MPEG-4 Parametric Audio Coding Tools," in *Proc. International Symposium on Circuits and Systems (ISCAS)*, 2000.

[96] T. Verma, *A Perceptually Based Audio Signal Model With Application to Scalable Audio Compression*, Thèse de doctorat, Stanford University, 2000.

[97] R. Heusdens, R. Vafin, and W.B. Kleijn, "Sinusoidal modeling using psychoacoustic-adaptive matching pursuits," *IEEE Signal Processing Letters*, vol. 9, no. 8, pp. 262–265, Aug. 2002.

[98] R. Boyer, *Modélisation et codage de signaux audio par extension du modèle sinusoïdal – Représentations compacte des signaux à variation rapides*, Thèse de doctorat, ENST, Paris, France, 2002.

[99] J. Jensen, R. Heusdens, and S.H. Jensen, "A Perceptual Subspace Approach for Modeling of Speech and Audio with Damped Sinusoids," *IEEE Trans. Speech and Audio Proc.*, 2003, To appear.

[100] L. Tancerel, S. Ragot, V.T. Ruoppila, and R. Lefebvre, "Combined speech and audio coding by discrimination," in *Proc. IEEE Workshop on Speech Coding*, Sep. 2000.

[101] P. Combescure et al., "A 16, 24, 32 kbit/s wideband speech coded based on ATCELP," in *Proc. ICASSP*, 1999, vol. 1, pp. 5–8.

[102] J. Schnitzler, J. Eggers, C. Erdmann, and Peter Vary, "Wideband speech coding using forward/backward adaptive prediction with mixed time/frequency domain excitation," in *Proc. IEEE Workshop on Speech Coding*, 1999, pp. 4–6.

[103] ITU-T Recommandation, "G.729 Annex E," Sep. 1998.

[104] J. Herre and J.D. Johnston, "Enhancing the Performance of Perceptual Audio Coders by Using Temporal Noise Shaping (TNS," in *Proc. 101st AES convention, Los Angeles*, 1996, Preprint 4384.

[105] J. Herre, "Temporal Noise Shaping (TNS, Quantization and Coding Methods in Perceptual Audio Coding: A Tutorial Introduction," in *Proc. AES 17th International Conference on High Quality Audio Coding*, 1999.

[106] G.D.T. Schuller, B. Yu, D. Huang, and B. Edler, "Perceptual Audio Coding Using Adaptive Pre- and Post-Filters and Lossless Compression," *IEEE Trans. Speech and Audio Proc.*, vol. 10, no. 6, Sep. 2002.

[107] T. Mirya et al., "A design transform coder for both speech and audio signals at 1 bit/sample," in *Proc. ICASSP*, 1997, vol. 2, pp. 1371–1374.

[108] H.S. Malvar, "Enhancing the performance of subband audio coders for speech signals," in *Proc. International Symposium on Circuits and Systems*, 1998, pp. 98–101.

[109] H. Taddei, *Codage hiérarchique faible retard 8–14.1–24 kbit/s pour les nouveaux réseaux et services*, Thèse de doctorat, Université de Rennes I, Rennes, France, 00.

[110] S. Ramprashad, "A Two-Stage Hybrid Embedded Speech/Audio Coding Structure," in *Proc. ICASSP*, 1998.

[111] ITU-T SG 16 (Multimedia Services and Systems) Temporary Document (WP3/16), "Question 20/16: Audio and wideband coding in public telecommunication networks – meeting report," Rapporteur, March 1997.

[112] ISO/IEC JTC1/SC29/WG11 MPEG, "Mpeg-4 call for proposals," N0997, Audio Sub group, July 1995.

[113] ISO/IEC JTC1/SC29/WG11 MPEG, "Call for proposals for new tools for audio coding," N3794, Audio Sub group, Jan. 2001.

[114] 3GPP TSG-SA4 Tdoc S4 (03)0014, "AMR-WB+ development work in SA4, Document for Discussion," TSG-SA4 25th meeting, San Francisco, CA, USA, Jan. 2003.

[115] T. Eriksson, J. Lindén, and J. Skoglund, "A safety-net approach for improved exploitation of speech correlations," in *Proc. Int. Conf. on Digital Signal Processing*, 1995, vol. 1, pp. 96–101.

[116] TSG-SA4 WG4, "Extended AMR-WB codec (AMR-WB+) targeted for packet-switched streaming and messaging," 3GPP Tdoc S4-030083, 2003.

[117] B. Bessette, R. Salami, C. Laflamme, and R. Lefebvre, "The Adaptive Multirate Wideband Speech Codec (AMR-WB)," *IEEE Trans. on Speech and Audio Processing*, vol. 10, no. 8, pp. 620–637, Nov. 2002.

[118] J.D. Gibson and K. Sayood, "Lattice Quantization," *Adv. Electon. Phys.*, vol. 72, pp. 259–331, 1988.

[119] M. Xie and J.-P. Adoul, "Embedded algebraic vector quantizers (EAVQ) with application to wideband speech coding," in *Proc. ICASSP*, 1996, vol. 1, pp. 240–243.

[120] J. Pan and T.R. Fischer, "Vector quantization-lattice vector quantization of speech LPC coefficients," in *Proc. ICASSP*, 1994, vol. 1, pp. 513–516.

[121] J. Pan, "Two-stage Vector Quantization-Pyramidal Lattice Vector Quantization and Application to Speech LSP Coding," in *Proc. ICASSP*, 1995, vol. 2, pp. 737–740.

[122] J. Pan, "Two-stage vector quantization-pyramidal lattice vector quantization and application to speech lsp coding," in *Proc. ICASSP*, 1996, vol. 2, pp. 737–740.

[123] M. Xie and J.-P. Adoul, "Fast and low-complexity LSF quantization using algebraic vector quantizer," in *Proc. ICASSP*, 1995, vol. 1, pp. 716–720.

[124] M. Xie and J.-P. Adoul, "Algebraic vector quantization of LSF parameters with low storage and computational complexity," *IEEE Trans. on Speech and Audio Proc.*, vol. 4, no. 3, pp. 234–239, May 1996.

[125] S. Ragot, J.-P. Adoul, R. Lefebvre, and R. Salami, "Low complexity LSF quantization for wideband speech coding," in *Proc. IEEE Workshop on Speech Coding*, 1999, pp. 22–24.

[126] S. Ragot, R. Lefebvre, R. Salami, and J.-P. Adoul, "Stochastic-algebraic wideband LSF quantization," in *Proc. ICASSP*, 2000, vol. II, pp. 1169–1172.

[127] M. Xie, *Quantification vectorielle algébrique et codage de parole en bande élargie*, Thèse de doctorat, Université de Sherbrooke, Québec, Canada, Février 1996.

[128] A.D. Subramanian and B.D. Rao, "PDF Optimized Parametric Vector Quantization of Speech Line Spectral Frequencies," *IEEE Trans. on Speech and Audio Processing*, vol. 11, no. 2, pp. 130–142, Mar. 2003.

[129] S. Ragot, H. Lahdili, and R. Lefebvre, "Wideband LSF quantization by generalized Voronoi codes," in *Proceedings of Eurospeech, Aalborg, Denmark*, Sep. 2001, pp. 2319–2322.

[130] G.D. Forney, "Coset codes. I. Introduction and geometrical classification," *IEEE Trans. Inform. Theory*, vol. 34, no. 5, pp. 1123–1151, Sep. 1988.

[131] L. Tancerel, "Discrimination parole/musique pour le codage universel de l'audio," M.S. thesis, Université de Sherbrooke, Québec, Canada, Déc. 1999.

[132] F. Labonté, "Étude, optimisation et implémentation d'un quantificateur vectoriel algébrique encastré dans un codeur audio hybride ACELP/TCX," M.S. thesis, Université de Sherbrooke, Québec, Canada, Avril 2001.

[133] H. Lahdili, "Mise en forme du bruit de codage dans la norme g.722.1 itu-t," M.S. thesis, Université de Sherbrooke, Québec, Canada, Juil. 2002.

[134] M. Ossonce, "Codage conjoint source/canal appliqué à un codeur hybride parole/audio," M.S. thesis, Université de Sherbrooke, Québec, Canada, en préparation.

[135] S. Ragot, "Design of Stochastic-Algebraic ISF Quantizers for ACELP.Live," Unpublished draft memo, University of Sherbrooke, QC, Canada, Aug. 2000.

[136] V.T. Ruoppila and S. Ragot, "Index assignment for predictive wideband LSF quantization," in *Proc. IEEE Workshop on Speech Coding*, Sept. 2000, pp. 108–110.

[137] S. Ragot, "Prédiction linéaire avec une résolution fréquentielle non-uniforme : théorie et applications," Unpublished draft memo, University of Sherbrooke, QC, Canada, Nov. 2001.

[138] S. Ragot, "Near-Transparent Switching for Hybrid Speech/Audio Coders," Unpublished draft memo, University of Sherbrooke, QC, Canada, Aug. 2000.

[139] S. Ragot, "Overview of the packetization strategy in AMR-WB+," Confidential report, VoiceAge Corp., Fev. 2003.

Chapitre 2

QUANTIFICATION PAR RÉSEAU DE POINTS

La quantification par réseau de points (*lattice quantization*) est une techique de quantification par contrainte, tout comme la quantification structurée en arbre, la quantification multi-étage, la quantification par produit cartésien (par sous-vecteurs) ou la quantification en treillis [1]. La contrainte porte ici sur le dictionnaire de quantification, dans le sens où celui-ci est défini comme un sous-ensemble d'un réseau de points. On ne s'intéresse ici qu'aux réseaux dits réguliers (*lattice*) associés à des empilements de sphères (*lattice sphere packings*), étudiés dans [2] ; on obtient ainsi une technique de quantification uniforme. L'objectif de ce chapitre est d'en présenter les aspects fondamentaux et d'en dresser un état de l'art (techniques, performances, etc.).

Ce chapitre est organisé comme suit. L'historique et les motivations de la quantification par réseau de points sont revus brièvement à la section 2.1. Les réseaux de points sont définis et étudiés à la section 2.2. Le problème de la recherche du plus proche voisin dans un réseau (infini) est explicité à la section 2.3. Les techniques classiques de quantification sont présentées à la section 2.4. L'analyse théorique des performances de la quantification par réseau de points est rappelée à la section 2.5, avant d'en examiner les caractéristiques réelles à la section 2.6. Ce chapitre est résumé

à la section 2.7

2.1 Bref historique et motivations de l'application des réseaux de points en quantification

L'étude fondamentale – en mathématiques – des réseaux de points et des empilements de sphères remonte au XVIII^{ième} siècle (Newton, Gauss, Minkowski, Hermite, etc.) [3]. Ces travaux sont restés inutilisés en communications jusqu'à ce que Shannon formule de façon géométrique le problème du codage sur un canal gaussien à bande limitée comme un problème d'empilement de sphères de bruit dans l'espace signal [4]. Mais c'est surtout Blake et DeBuda qui ont ouvert la voie à l'utilisation des réseaux de points en communications en construisant des codes de canal à l'aide de ces structures [5, 6].

L'application des réseaux de points à la quantification vectorielle a d'abord été étudiée sur le plan théorique. En particulier, Gersho a observé en 1979 qu'en dimension élevée le quantificateur optimal pour une contrainte d'entropie a la structure d'un réseau de points [7]; cette conjecture de Gersho n'a cependant jamais été démontrée, bien qu'elle soit généralement considérée exacte. Ce résultat n'est que l'extension des travaux de Gish, Pierce et Wood sur la quantification scalaire [8, 9], qui n'ont été démontrés rigoureusement que récemment. Gersho a aussi défini des quantificateurs par réseau de points en dimensions 2, 3 et 4. Il a également proposé d'étendre les techniques non-linéaires de compression-expansion du cas scalaire au cas vectoriel [7], à partir de la méthode de Bennett [10]. En effet, les sources naturelles de distributions généralement non-uniformes ne sont pas adaptées à une quantification uniforme. La généralisation multi-dimensionnelle de la fonction de compression est néanmoins difficile [7, 11]. Ce dernier résultat de Gersho sur la compression multi-dimensionnelle a en fait ralenti pour un temps l'application des réseaux de points en quantification.

Les recherches pratiques sur la quantification par réseau de points ont réellement débuté à partir des travaux de Sloane et Conway [12, 13, 14, 15] de 1981 à 1983, avec en particulier une tabulation de codes sphériques et la publication d'algorithmes rapides de recherche du plus proche voisin

et d'indexation. L'étude des réseaux de points en quantification était alors motivée par le fait que les techniques scalaires n'atteignent pas la borne débit-distorsion, tandis que la quantification vectorielle, sous sa forme non structurée (LBG) introduite en 1980 [16], requiert un apprentissage généralement coûteux et possède une complexité exponentielle en fonction du débit et de la dimension. La quantification vectorielle par réseau de points était perçue comme l'alternative principale à la quantification vectorielle de type LBG, permettant de se rapprocher de la borne débit-distorsion, tout en ayant une complexité potentiellement plus faible [1].

Les premiers algorithmes de quantification par réseau de points sont apparus en 1984, avec la publication d'une technique de quantification adaptée à des sources uniformes par Sayood, Gibson et Rost [17] et le développement d'algorithmes efficaces de quantification sphérique par Adoul [18]. La quantification pyramidale a été introduite par Fischer [19] en 1986. D'autres techniques ont suivi ces premiers développements. La quantification par réseau de points n'a donc été appliquée qu'à cette date à des problèmes réels (codage d'images par transformée [17] et codage de la parole [18]), alors que les réseaux de points étaient déjà utilisés en modulation numérique depuis au moins 1981 [20, 21]. La quantification par réseau de points est par la suite devenue relativement populaire en codage d'images et vidéo. Par contre, son application est restée plus anecdotique en codage de parole et audio, où d'autres outils de codage sont généralement utilisés – par exemple, les codes impulsionnels à permutations, la quantification scalaire avec codage de Huffman et la quantification vectorielle multi-étage et par produit cartésien.

2.2 Réseaux de points

On s'intéresse ici aux notions sur les réseaux de points pertinentes en quantification. Des aspects connexes, tels que les problèmes d'empilement de sphères ou de couverture d'espace et les formes quadratiques sont discutés dans [2, 3].

2.2.1 Définitions de base

Réseau régulier : Un réseau régulier Λ est défini ici dans $\mathbb{R}^N$ comme l'ensemble des points qui s'obtiennent par combinaison linéaire de M vecteurs de base linéairement indépendants $\mathbf{v}_1$, $\mathbf{v}_2$, ..., $\mathbf{v}_M$ à coefficients de proportionalité entiers, soit :

$$\Lambda = \left\{ \mathbf{x} \in \mathbb{R}^N : \mathbf{x} = \sum_{i=1}^{M} k_i \mathbf{v}_i : \quad k_i \in \mathbb{Z}, \mathbf{v}_i \in \mathbb{R}^N, \quad \forall i = 1, 2, \cdots, M \right\}. \tag{2.1}$$

Le nombre M de vecteurs de base définit le rang (ou dimension) de Λ – on ne considère ici que des réseaux de rang plein (i.e. $M = N$). Un exemple arbitraire de réseaux régulier de points en dimension 2 est montré à la figure 2.1.

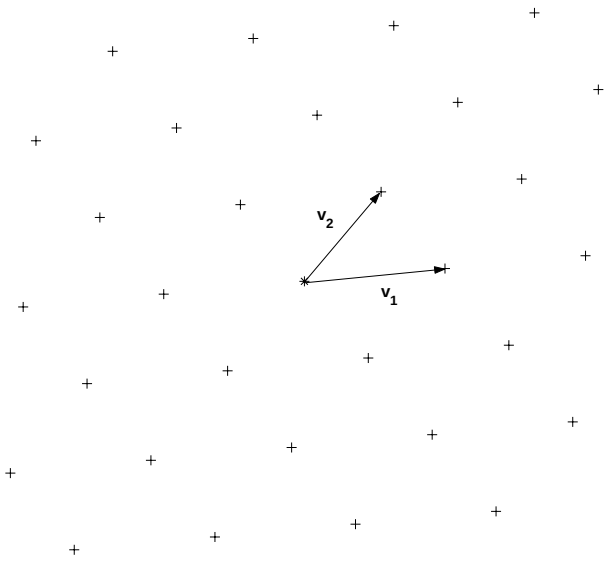


Figure 2.1: Exemple de réseau régulier de points.

Un arrangement discret de points dans l'espace $\mathbb{R}^N$ (i.e. un réseau de points) est en général irrégulier. La spécificité des réseaux réguliers tient donc à leur structure algébrique linéaire : un réseau régulier est un arrangement de points qui possède une structure de groupe additif – une telle structure est appelée $\mathbb{Z}$-module en mathématiques. On s'intéresse ici plus particulièrement

aux réseaux associés à des empilements multi-dimensionnels de sphères [2], qui conduisent à une densité de points uniforme dans l'espace. En dimension 2, un empilement de sphères consiste à placer régulièrement des disques disjoints de même diamètre dans le plan. Les centres de ces sphères (ou disques) forment un réseau de points. On ne retient ici que les empilements dont les centres forment un réseau régulier et dans lesquels les sphères se touchent [3]. Des exemples d'empilements en dimension 2 associés des réseaux réguliers sont montrés à la figure 2.2.

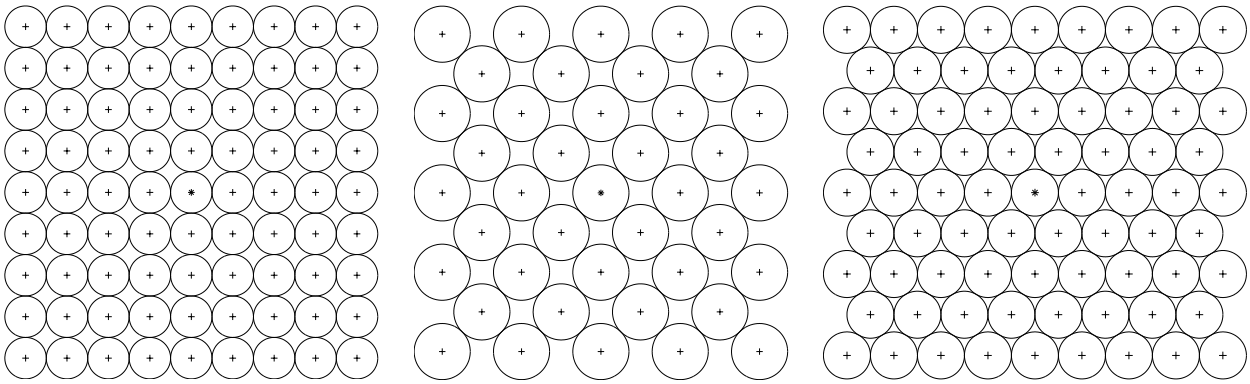


Figure 2.2: Exemples d'empilements de sphères en dimension 2.

Les réseaux de points trouvent de nombreuses applications en communications : quantification vectorielle, modulation codée et cryptographie [2]. Ils sont étudiés de façon systématique et catalogués dans [2]. Leur étude mathématique constitue une branche de la théorie des nombres remontant à Gauss et Minkowski ; ils font également l'objet d'études en chimie (cristallographie), en physique théorique, etc.

Matrice génératrice : Les vecteurs de base d'un réseau régulier Λ peuvent être regroupés dans une matrice génératrice

$$M(\Lambda) = \begin{bmatrix} \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_N \end{bmatrix} = \begin{bmatrix} v_{11} & \cdots & v_{1N} \\ \vdots & & \\ & & \vdots \\ v_{N1} & \cdots & v_{NN} \end{bmatrix}, \tag{2.2}$$

où v_{ij} est la j-ième composante de $\mathbf{v}_i$, de sorte que

$$\Lambda = \left\{ \mathbf{x} = \mathbf{k}M(\Lambda) : \mathbf{k} \in \mathbb{Z}^N \right\}. \tag{2.3}$$

On adopte donc ici comme convention que les lignes de $M(\Lambda)$ définissent la base de Λ – la convention colonne est néanmoins plus courante.

Si $M(\Lambda)$ génère Λ, toute matrice de la forme $TM(\Lambda)$, où T est une matrice entière orthogonale ($\det T = \pm 1$), génère également Λ. Une telle matrice T est dit unimodulaire entière ; elle transforme $\mathbb{Z}^N$ en lui-même.

Réduction de base : La longueur d'un vecteur $\mathbf{v}$ de $\mathbb{R}^N$ est définie comme la norme euclidienne $\|\mathbf{v}\|$. Dans certains cas, on cherche parmi toutes les bases possibles de Λ une base constituée des vecteurs les plus courts (pour un critère donné); ce problème est appelé réduction de base. Différents critères et algorithmes de réduction (Minkovski, Korkine-Zolotareff, Lenstra-Lenstra-Lovász) sont discutés par exemple dans [22].

Matrice de Gram et réseau dual : La matrice de Gram de Λ, définie par

$$A = M(\Lambda)M(\Lambda)^t, \tag{2.4}$$

caractérise complètement Λ. Cette matrice est indépendante de la base de Λ ; si elle est à coefficients entiers, le réseau Λ est dit intégral [2]. Cette matrice permet en particulier de calculer une matrice génératrice $M(\Lambda)$ triangulaire inférieure par factorisation (décomposition) de Cholesky sous la forme $A = LL^t$, où $L = M(\Lambda)$ est triangulaire inférieure.

Le réseau dual Λ^* d'un réseau Λ est

$$\Lambda^* = \left\{ \mathbf{x} \in \mathbb{R}^N \, : \mathbf{x}\mathbf{y}^t \in \mathbb{Z} \;\; \forall \mathbf{y} \in \Lambda \right\}. \tag{2.5}$$

Cette propriété est pertinente ici, car les réseaux intéressants en quantification sont souvent les duaux des réseaux les plus denses [2]. Si A est la matrice de Gram de Λ, A^{-1} est la matrice de Gram de Λ^*. Pour un réseau Λ de rang plein, une matrice génératrice de Λ^* est $(M(\Lambda)^t)^{-1}$.

2.2.2 Exemples de réseaux réguliers importants

Le réseau de points le plus simple est sans conteste $\mathbb{Z}^N$, spécifié par une matrice génératrice identité. Ce réseau, appelé aussi grille cartésienne ou réseau cubique, est associé à une quantification scalaire uniforme (dimension par dimension, avec un même pas de quantification dans chaque dimension). Parmi les autres exemples simples de réseaux, on trouve les familles A_N ($N \geq 2$), D_N ($N \geq 2$) et $2D_N^+$ (N pair ≥ 4).

Le réseau A_N est défini sur un hyperplan de dimension N dans $\mathbb{R}^{N+1}$ comme

$$A_N = \left\{ x \in \mathbb{Z}^{N+1} | \sum_{i=1}^{N+1} x_i = 0 \right\}. \tag{2.6}$$

Ce réseau peut être défini directement en dimension N après rotation adéquate de l'hyperplan d'équation $x_1 + x_2 + \cdot + x_N = 0$ (pour éliminer une dimension). Ainsi, le réseau A_2 peut être spécifié de façon équivalente en dimension 2 par $\mathbf{v}_1 = \left[1/2, \sqrt{3}/2\right]$ et $\mathbf{v}_2 = [1, 0]$. On trouve plus de détails à ce sujet, en particulier la rotation de l'hyperplan d'équation $x_1 + x_2 + \cdot + x_N = 0$, dans [23].

Le réseau D_N est défini comme :

$$D_N = \left\{ \mathbf{x} \in \mathbb{Z}^N | \sum_{i=1}^{N} x_i \text{ est pair} \right\} \tag{2.7}$$

C'est un sous-ensemble de $\mathbb{Z}^N$ puisque $\mathbb{Z}^N = D_N \cup \{D_N + (1, 0, \cdots, 0)\}$, où

$$D_N + (1, 0, \cdots, 0) = \left\{ \mathbf{x} \in \mathbb{Z}^N : \mathbf{x} = (d_1 + 1, d_2, \cdots, d_N) : (d_1, d_2, \cdots, d_N) \in D_N \right\}. \tag{2.8}$$

Le réseau D_4 est appelé réseau de Schläfli.

Enfin, le réseau $2D_N^+$ (N pair ≥ 4) est défini par

$$2D_N^+ = 2D_N \bigcup \{2D_N + (1, \cdots, 1)\} \tag{2.9}$$

où

$$2D_N = \left\{ \mathbf{x} \in \mathbb{Z}^N : \mathbf{x} = (2d_1, 2d_2, \cdots, 2d_N) : (d_1, d_2, \cdots, d_N) \in D_N \right\}. \tag{2.10}$$

Le réseau $2D_8^+$ est appelé réseau de Gosset.

Ces réseaux, ainsi que d'autres, sont définis par des matrices génératrices ou matrices de Gram données à l'annexe 2.A. Les réseaux $\mathbb{Z}^2$, D_2 et A_2 sont illustrés à la figure 2.3.

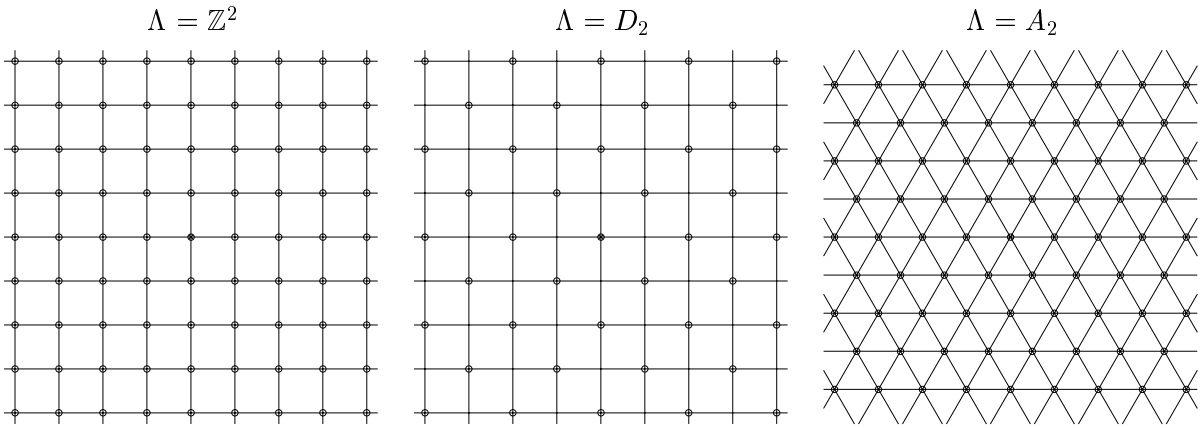


Figure 2.3: Réseaux $\mathbb{Z}^2$, D_2 et A_2 (avec matrices génératrices).

2.2.3 Propriétés géométriques

Réseaux équivalents (ou géométriquement similaires) : Des réseaux Λ et Λ' sont dits équivalents ($\Lambda \cong \Lambda'$) s'ils diffèrent uniquement par rotation ou dilatation (d'une facteur < 1 ou > 1). La matrice génératrice $M(\Lambda')$ est de la forme $M(\Lambda') = cTM(\Lambda)B$, où c est une constante, B une matrice de rotation/réflexion orthogonale réelle ($B^tB =$ identité) et T est une matrice à coefficients entiers unimodulaire. Par exemple, les réseaux $\mathbb{Z}^2$ et D_2 sont de réseaux géométriquement similaires. On donne à la figure 2.4 un exemple où $\Lambda = A_2$ est tourné de $\pi/4$ et dilaté par un facteur $\sqrt{2}$.

Parallélotope de base et régions de Voronoï : Le parallalélotope de base d'un réseau de point Λ est la région $\Pi(\Lambda)$ définie par :

$$\Pi(\Lambda) = \left\{ \mathbf{x} = \mathbf{u}M(\Lambda) | \mathbf{u} \in \mathbb{R}^N, 0 \leq u_i < 1 \ \ \forall i = 1, 2, \cdots, N \right\}. \tag{2.11}$$

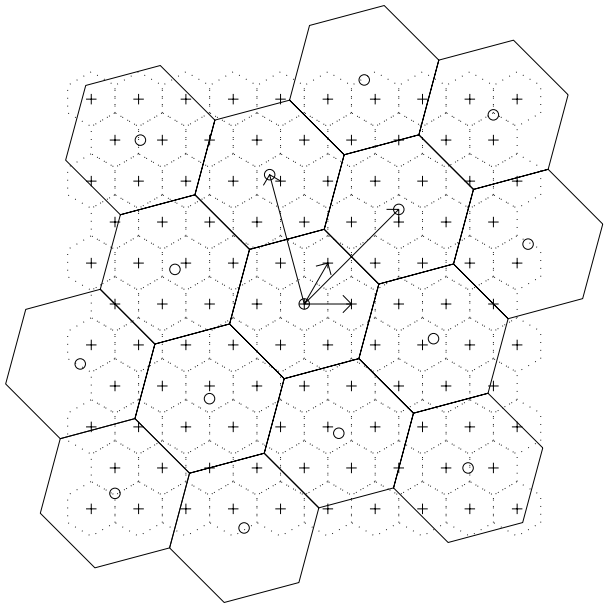


Figure 2.4: Exemples de réseaux géométriquement similaires : $\Lambda = A_2$ (+) et $\Lambda' = A_2$ tourné de $\pi/4$ et mis à l'échelle (o) – les vecteurs de chaque base sont indiqués par des vecteurs.

Ce parallélotope dépend de la base adoptée pour Λ. La région de Voronoï relative à l'origine est $V(\Lambda)$ définie par :

$$V(\Lambda) = \left\{ \mathbf{x} \in \mathbb{R}^N \mid ||\mathbf{x}||^2 < ||\mathbf{x} - \mathbf{y}||^2 \text{ pour tout } \mathbf{y} \in \Lambda - \{\mathbf{0}\} \right\} \tag{2.12}$$

où $\|.\|$ est la norme euclidienne. Dans un réseau régulier de points, les régions de Voronoï sont toutes congruentes, et la région de Voronoï relative à $\mathbf{y}$ est $V(\Lambda) + \mathbf{y}$. En terme de volumes, $\mathrm{Vol}(V(\Lambda)) = \mathrm{Vol}(\Pi(\Lambda)) = |\det M(\Lambda)|$ – en supposant que le réseau Λ est de rang plein. Les régions $\Pi(\Lambda)$ et $V(\Lambda)$ centrées en chaque point de Λ permettent d'obtenir un pavage régulier de l'espace $\mathbb{R}^N$.

Les régions de Voronoï des réseaux $\mathbb{Z}^2$, D_2 et A_2 sont illustrées à la figure 2.5.

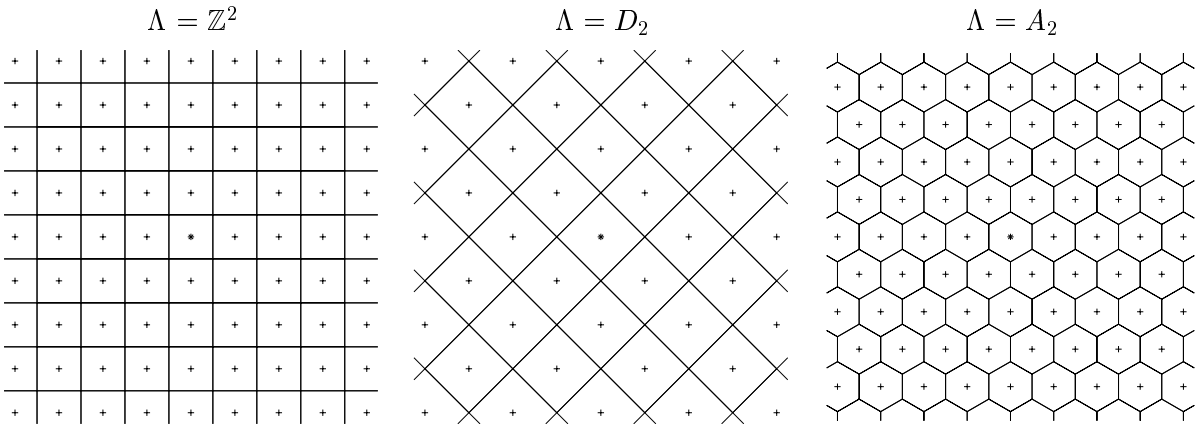


Figure 2.5: Réseaux $\mathbb{Z}^2$, D_2 et A_2 : frontières des régions de Voronoï.

Géométrie de $V(\Lambda)$ - rayon d'empilement, distance minimale, trous et rayon de couverture R, vecteurs pertinents : La distance minimale d_{min} d'un réseau Λ est définie par :

$$d_{min} = \min_{\mathbf{x},\mathbf{y}\in\Lambda \text{ tels que } \mathbf{x}\neq\mathbf{y}} \|\mathbf{x}-\mathbf{y}\| \tag{2.13}$$

où $\|.\|$ désigne la norme euclidienne. Cette quantité généralise la notion de pas de quantification scalaire.

Le rayon d'empilement est $\rho = d_{min}/2$ [2] ; il correspond au rayon des sphères identiques dans l'empilement associé à Λ. Le nombre de points de contact τ est le nombre de sphères en contact avec une sphère donnée dans l'empilement associé à Λ.

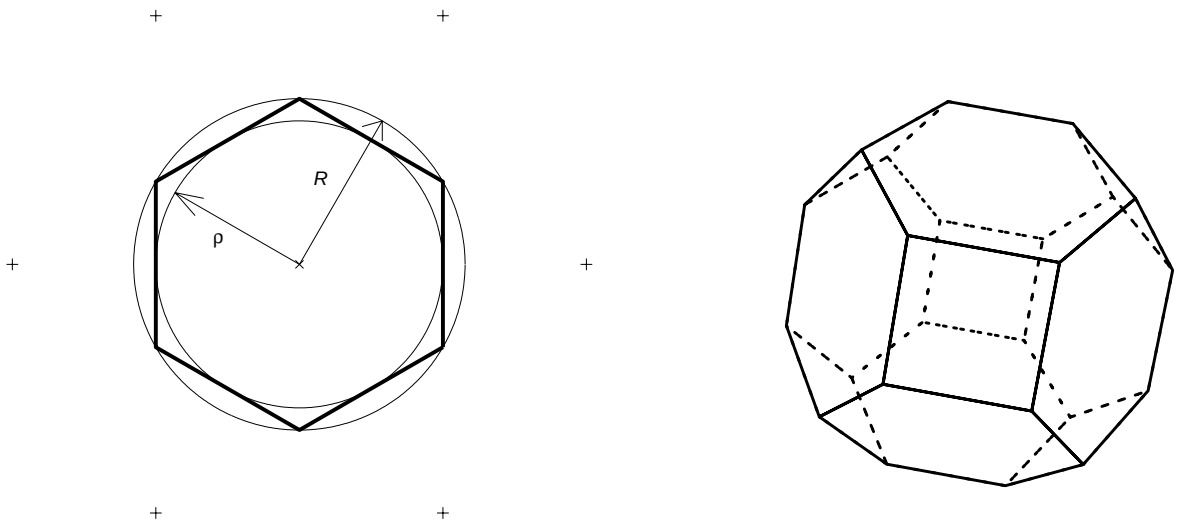
Un trou de Λ est un point de $\mathbb{R}^n$ dont la distance à Λ est un maximum local [2]. La distance maximale à Λ de tous les trous de Λ est appelée le rayon de couverture R de Λ ; ce rayon correspond au rayon identique des sphères dans la couverture de l'espace associée à Λ. Un trou à distance R de Λ est appelé un trou profond.

La région de Voronoï $V(\Lambda)$ a pour sommets les trous de Λ relatifs à l'origine. De plus, $V(\Lambda)$ est circonscrite à l'intérieur de la sphère centrée de rayon R ; cette sphère passe par les trous profonds de Λ relatifs à $\mathbf{0}$.

La frontière de $V(\Lambda)$ est constituée de portions d'hyperplans affines qu'on appelle des faces.

La région $V(\Lambda)$ est un polytope, c'est-à-dire une intersection bornée de demi-espaces. Un point $\mathbf{p}$ de Λ tels que l'hyperplan affine entre $\mathbf{0}$ et $\mathbf{p}$ contient une face de $V(\Lambda)$ est appelé vecteur pertinent de Λ. L'algorithme de "taille du diamant" de [24] permet de calculer les caractéristiques de $V(\Lambda)$ (nombre de faces, nombre de sommets, nombre de points de contact, rayons d'empilement et de couverture, etc.) – pour des réseaux de faible dimension ($N \leq 8$).

Ces propriétés sont illustrées à la figure 2.6 dans le cas des réseaux A_2 et A_3^*.



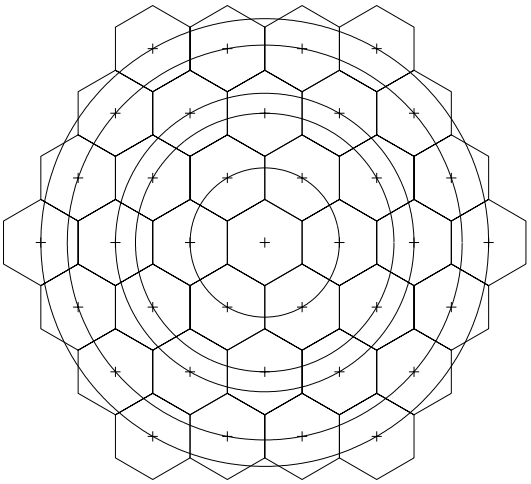
(a) $V(A_2)$: faces, vecteurs pertinents (+)

(b) $V(A_3^*)$ après rotation adéquate pour faciliter la visualisation

Figure 2.6: Régions de Voronoï $V(A_2)$ et $V(A_3^*)$.

Énumération des points par orbite : séries Θ, ν, etc. Les points d'un réseau sont définis sur des orbites sphériques concentriques autour de l'origine. Cette propriété est illustrée à la figure 2.7 (a), qui représente les 6 premières orbites sphériques non vides de A_2.

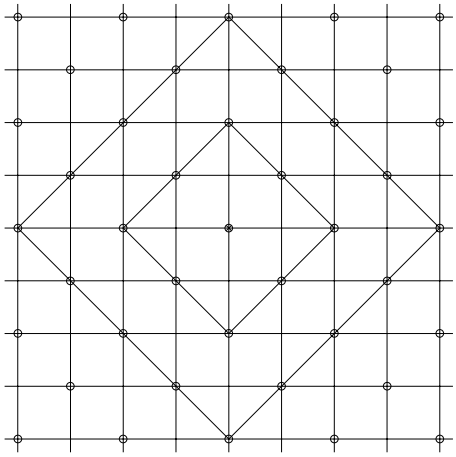
L'énumération des points de Λ par orbite sphérique peut être calculée à partir de la matrice



(a) Orbites sphériques de rayon 0 à 3 dans A_2

série Θ avec m=0,1,3,4,7,9 :

$\Theta_{A_2}(z) = 1 + 6q + 6q^3 + 6q^4 + 12q^7 + 6q^9 \cdots$



(b) Orbites pyramidales de rayon 0 à 3 dans D_2

série ν : $\nu_{D_2}(z) = 1 + 8q + 16q^3 + \cdots$

Figure 2.7: Exemples de décomposition par orbite sphérique et pyramidale.

génératrice $M(\Lambda)$. Elle est donnée de façon compacte par la série formelle Θ de Λ définie par [2] :

$$\Theta_\Lambda(z) = \sum_{\mathbf{x}\in\Lambda} q^{\|\mathbf{x}\|^2} = \sum_{m=0}^{\infty} N_m q^m \tag{2.14}$$

avec $q = e^{\pi i z}$ et $i = \sqrt{-1}$. Les coefficients N_m de cette série correspondent au nombre de points de Λ sur chaque orbite sphérique de Λ centrée sur l'origine de rayon $\sqrt{m}$. Par exemple, les premiers termes de la série $\Theta_{A_2}(z)$ sont $1 + 6q + 6q^3$, car la sphère 0 de rayon 0 comprend un seul point (l'origine), la sphère 1 de rayon 1 comprend 6 points, la sphère de rayon 3 comprend 6 points, etc.

La série Θ est déterminée de façon unique pour Λ donné, mais des réseaux différents peuvent avoir la même série Θ. Par suite, la série débute toujours par :

$$\Theta_\Lambda(z) = 1 + \tau q + \cdots \tag{2.15}$$

Si le réseau est intégral, l'indice m est entier car les distances entre points dans le réseau sont entières [2]. En général, si l'énunération est réalisée relativement à un trou profond de Λ ou un autre point remarquable, cette distribution des orbites de Λ change.

L'énumération des points par séries formelles est généralisée dans [25, 26, 27, 28] (séries ν et θ modifiées, etc.). Les séries ν correspondent à une énumération des points de Λ non plus par orbite sphérique mais par orbite pyramidale, comme celles montrées à la figure 2.7 (b) ; dans le cas des séries ν, l'énumération dépend de la base de Λ [25].

2.2.4 Propriétés algébriques

Cosets, sous-réseaux et partitions : Un coset d'un réseau Λ est $\Lambda + \mathbf{c}$,où $c \in \mathbb{R}^N$. Le coset $\Lambda + \mathbf{c}$ est géométriquement un translaté de Λ. Des exemples de cosets sont montrés à la figure 2.8 pour $\Lambda = 2\mathbb{Z}^2$.

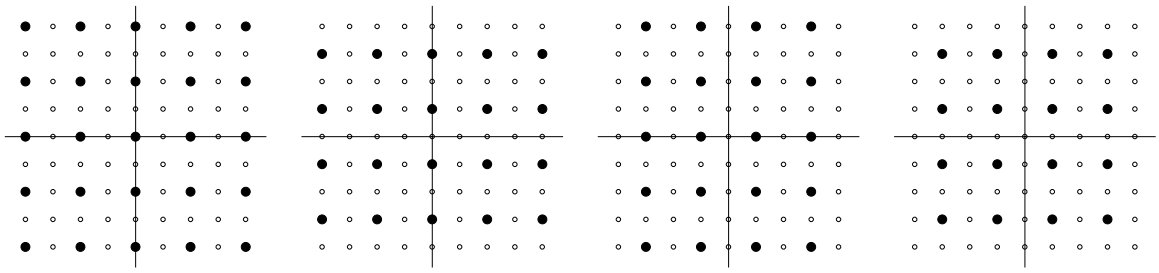


Figure 2.8: Cosets de $2\mathbb{Z}^2$.

Un sous-réseau Λ' de Λ est un sous-ensemble de Λ doté de la structure de réseau régulier (i.e. un sous-groupe du groupe additif Λ). Par exemple, en dimension 2, $2\mathbb{Z}^2$ est un sous-réseau de $\mathbb{Z}^2$.

Un tel sous-réseau induit une partition de Λ en classes d'équivalence. La relation d'équivalence $\mathcal{R}$ sur Λ, considérée ici, est définie par :

$$\mathbf{x}\ \mathcal{R}\ \mathbf{y} \Leftrightarrow \mathbf{x} - \mathbf{y} \in \Lambda', \tag{2.16}$$

où $\mathbf{x}$ et $\mathbf{y}$ désignent des points de Λ. Chaque classe d'équivalence (suivant $\mathcal{R}$) est en fait un coset de Λ' [29]. L'ensemble de ces classes d'équivalence définit la partition de Λ, notée Λ/Λ'. En prenant un élément de chaque classe, on obtient un ensemble de représentants de cosets, noté $[\Lambda/\Lambda']$. Tout

point $\mathbf{x}$ de Λ peut alors être décomposé sous la forme $\mathbf{x}' + \mathbf{c}$, où $\mathbf{x}' \in \Lambda'$ et $\mathbf{c} \in [\Lambda/\Lambda']$. Cette décomposition peut être résumée par la formule :

$$\Lambda = \Lambda' + [\Lambda/\Lambda']. \tag{2.17}$$

Le nombre d'éléments dans $[\Lambda/\Lambda']$ est noté $|\Lambda/\Lambda'|$ et correspond mathématiquement à la cardinalité de la partition Λ/Λ'.

Par exemple, en dimension 1, le réseau $m\mathbb{Z}$ avec m entier ≥ 2 est un sous-réseau de $\mathbb{Z}$. Un choix naturel de représentants de coset est $[\mathbb{Z}/m\mathbb{Z}] = \{0, 1, \cdots, m-1\}$ – par suite, $|\mathbb{Z}/m\mathbb{Z}| = m$. Le réseau $\mathbb{Z}^2$ comprend 4 cosets de $2\mathbb{Z}^2$, tels que montrés à la figure 2.8 ; ainsi $|\mathbb{Z}^2/2\mathbb{Z}^2| = 4$ et on peut choisir $[\mathbb{Z}^2/2\mathbb{Z}^2] = \{(0,0), (0,1), (1,0), (1,1)\}$.

Constructions algébriques de Forney : La décomposition de réseaux de points en cosets permet de construire des réseaux en garantissant une certaine distance minimale. Des constructions générales sont présentées dans [30] (quadrature, cubature, double quadrature ou quadrature à 2 niveaux de partitions, etc.). Par exemple, pour une partition Λ/Λ', la construction en quadrature est définie par :

$$|\Lambda/\Lambda'|^2 = \left\{(\lambda'_1 + \mathbf{c}, \lambda'_2 + \mathbf{c}) | \lambda'_1, \lambda'_2 \in \Lambda', \mathbf{c} \in [\Lambda/\Lambda']\right\} \tag{2.18}$$

La distance minimale est alors garantie à au moins $min(d_{min}(\Lambda'), 2d_{min}(\Lambda))$. Les constructions de Forney conduisent par ailleurs à des matrices génératrices de réseaux ayant une structure intuitive. Elles complétent les constructions classiques de réseaux (constructions A, B, etc. [2]).

Réseaux binaires et codes correcteurs : Un réseau de points Λ est binaire s'il est entier (i.e. Λ est un sous-réseau de $\mathbb{Z}^N$) et s'il admet comme sous-réseau $2^p\mathbb{Z}^N$ (avec $p \in \mathbb{N}$) [29, p. 1130]. Les réseaux binaires ont la particularité d'admettre comme représentants de cosets des codes correcteurs d'erreurs (le plus souvent binaires). On trouve ainsi [30] :

$$D_4 = 2\mathbb{Z}^4 + (4,3,2) \tag{2.19}$$

$$RD_4 = 2\mathbb{Z}^4 + (4,1,4) \tag{2.20}$$

$$
\begin{aligned}
E_8 &= 2\mathbb{Z}^8 + (8,4,4) && (2.21)\\
RE_8 &= 4\mathbb{Z}^8 + 2(8,7,2) + (8,1,8) && (2.22)\\
\Lambda_{16} &= 4\mathbb{Z}^{16} + 2(16,15,2) + (16,5,8) && (2.23)\\
R\Lambda_{16} &= 4\mathbb{Z}^{16} + 2(16,11,4) + (16,1,16) && (2.24)\\
\Lambda_{24} &= 4\mathbb{Z}^{24} + 2(24,18,4) + (24,6,16) && (2.25)\\
R\Lambda_{24} &= 8\mathbb{Z}^{24} + 4(24,23,2) + 2(24,12,8) + (24,1,24)^* && (2.26)\\
\Lambda_{32} &= 4\mathbb{Z}^{32} + 2(32,26,4) + (32,6,16) && (2.27)\\
R\Lambda_{32} &= 8\mathbb{Z}^{16} + 4(32,31,2) + 2(32,16,8) + (32,1,32) && (2.28)
\end{aligned}
$$

Ces formules suivent la notation compacte de Forney [30]. Dans ces formules, les termes de la forme (n,k,d) désignent des codes correcteurs en bloc de longueur n de dimension k et de distance minimale d – un code (binaire) (n,k,d) comprend donc 2^k mots de n bits. Par exemple, le code binaire $(4,3,2)$ – appelé code à parité – comprend les 8 mots de 4 bits ayant un nombre pair de 1:

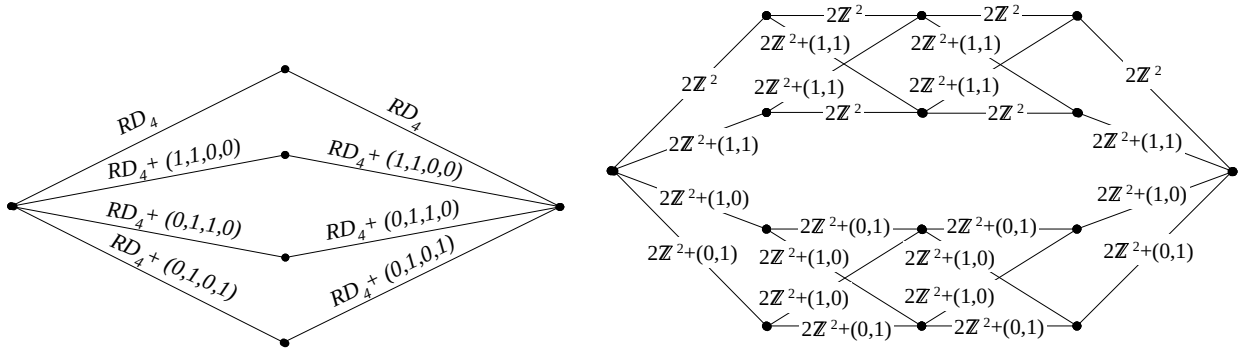
$$
(4,3,2) = \begin{cases}
(0,0,0,0)\\
(0,0,1,1)\\
(0,1,0,1)\\
(1,0,0,1)\\
(0,1,1,0)\\
(1,0,1,0)\\
(1,1,0,0)\\
(1,1,1,1)
\end{cases} \qquad (2.29)
$$

L'addition désigne une translation de réseaux de points. L'addition $D_4 = 2\mathbb{Z}^4 + (4,3,2)$ signifie que le réseau D_4 comprend les 8 translations possibles de $2\mathbb{Z}^4$ par un élément de $(4,3,2)$, soit

$$
D_4 = \left\{ 2\mathbf{z} + \mathbf{c} | \mathbf{z} \in \mathbb{Z}^2, \mathbf{c} \in (4,3,2) \right\}. \qquad (2.30)
$$

La translation $\mathbf{c} \in (4,3,2)$ peut prendre comme valeur $(0,0,0,0)$, $(1,1,1,1)$ ou une permutation de $(1,1,0,0)$. Ces relations entre réseaux de points et codes correcteurs sont connues depuis les travaux de Leech, Conway et Sloane, Barnes et Wall, etc [2, 31].

Treillis : La structure de cosets d'un réseau peut être représentée sous la forme d'un treillis [30]. Cette représentation conduit à des algorithmes efficaces de recherche du plus proche voisin [30]. Des exemples de treillis sont donnés à la figure 2.9 pour le réseau E_8.



(a) quadrature $|D_4/RD_4|^2$ (b) double quadrature $|\mathbb{Z}^2/R\mathbb{Z}^2/2\mathbb{Z}^2|^4$

Figure 2.9: Treillis du réseau E_8.

2.2.5 Moment normalisé d'ordre 2

En quantification à débit élevé et sous certaines hypothèses additionnelles, la quantification par réseau de points se ramène à approximer la distribution de la source comme étant uniforme par morceaux. Dans chaque région de Voronoï, l'erreur quadratique moyenne est alors proportionnelle à la distorsion moyenne résultant de l'arrondi d'une source uniforme dans $V(\Lambda)$ au centre de $V(\Lambda)$ (l'origine). Cette distorsion est donnée par $1/vol(V(\Lambda)) \int_{V(\Lambda)} \|\mathbf{x}\|^2 \, d\mathbf{x}$. Afin de comparer a priori la performance de différents réseaux (réguliers), cette erreur peut être normalisée par la dimension de quantification N et un facteur $vol(V(\Lambda))^{2/N}$. Dans ces conditions, la distorsion moyenne devient

$$G(\Lambda) = \frac{1}{N} \frac{1}{vol(V(\Lambda))^{1+2/N}} \int_{V(\Lambda)} \|\mathbf{x}\|^2 \, d\mathbf{x}. \tag{2.31}$$

Le facteur $vol(V(\Lambda))^{2/N}$ est justifié à l'annexe 2.B. La quantité $G(\Lambda)$ est sans dimension, dans le sens où elle invariante à la dilatation de Λ. Cette quantité est appelée moment normalisé d'ordre 2 ou constante de quantification de Λ. Elle représente l'erreur granulaire de quantification dans Λ (pour une source uniforme codée sans surcharge).

Valeurs de $G(\Lambda)$ pour les réseaux importants en dimension $N \leq 24$

Le moment normalisé d'ordre 2, $G(\Lambda)$, d'un réseau Λ en dimension N, est minoré par la constante de quantification d'une N-sphère (i,e. une sphère en dimension N) de même volume que $V(\Lambda)$ [32] :

$$G(\Lambda) \geq \frac{\Gamma(\frac{N}{2}+1)^{2/N}}{(N+2)\pi}. \tag{2.32}$$

Cette borne inférieure est appelée *sphere bound* en anglais. Plus la région de Voronoï $V(\Lambda)$ s'approche d'une $N-$sphère, plus $G(\Lambda)$ s'approche de cette borne. La quantité $G(\Lambda)$ résume donc le caractère sphérique de $V(\Lambda)$. Une borne plus précise (minoration plus forte) est donnée dans [33]. On sait de plus que :

$$\frac{\Gamma(\frac{N}{2}+1)^{2/N}}{(N+2)\pi} \longrightarrow \frac{1}{2\pi e} = 0.0585498 \text{ quand } N \to +\infty. \tag{2.33}$$

Les valeurs de $G(\Lambda)$ du tableau 2.1 sont tirées de [14, 2]. La définition récursive de J_N pour A_N^* se trouve dans [14] – en particulier, $J_N = 0, \frac{1}{12}, \frac{5}{18}, \frac{19}{32}, \cdots$ pour $N = 0, 1, 2, 3, \ldots$. Lorsqu'une formule analytique n'est pas disponible pour $G(\Lambda)$, la valeur de $G(\Lambda)$ peut être estimée facilement à l'aide de la méthode d'intégration de Monte Carlo. Un exemple est donné à l'annexe 2.C.

Le gain granulaire est défini par

$$\gamma_g(\Lambda) = 10 \log_{10} \frac{G(\mathbb{Z})}{G(\Lambda)}; \tag{2.34}$$

il est majoré à 1.53 dB (borne de la N-sphère). Ce gain représente la réduction d'erreur quadratique moyenne lorsqu'une source uniforme est quantifiée par Λ au lieu de $\mathbb{Z}$.

Le moment normalisé d'ordre 2 a été utilisé dans [34] comme critère d'optimisation numérique de réseaux réguliers (par recherche par gradient sur les composantes de la base).

Choix d'un réseau de points

Le choix du réseau de points dépend de la source, du critère de distorsion et du débit. Pour l'erreur quadratique, on peut distinguer deux cas.

TABLEAU 2.1: Indice de quantification des réseaux de points importants en dimensions $n \leq 24$.

Réseau	$G(\Lambda)$	$\gamma_g(\Lambda)$ (dB)
$\mathbb{Z}^N$	$\frac{1}{12}$	0
A_N $(N \geq 2)$	$\frac{1}{(N+1)^{1/N}} \left(\frac{1}{12} + \frac{1}{6(N+1)} \right)$	0.16 $(N = 2)$
A_N^* $(N \geq 2)$	$\frac{J_{N+1}}{N(N+1)^{1-1/N}}$	0.25 $(N = 3)$
D_N $(N \geq 4)$	$\frac{1}{2^{2/N}} \left(\frac{1}{12} + \frac{1}{2N(N+1)} \right)$	0.36 $(N = 4)$
D_5^*	$\frac{2641}{23040\, 2^{3/5}} = 0.0755654$	0.42
E_6^*	$\frac{12619\, 3^{1/6}}{204120} = 0.0742437$	0.50
E_7^*	$\frac{21361\, 2^{1/7}}{322560} = 0.0731165$	0.56
E_8, RE_8	$\frac{929}{12960} = 0.0716821$	0.65
K_{12}	0.070100	0.75
Λ_{16}, $R\Lambda_{16}$	0.068299	0.86
Λ_{24}, $R\Lambda_{24}$	0.065771	1.02

Choix à haut débit : D'après la théorie débit-distorsion à haut débit [2], l'erreur de quantification se réduit essentiellement à l'erreur granulaire, laquelle est indépendante de la source et fonction de $G(\Lambda)$. En effet à haut débit, la distribution de la source est uniforme par morceaux. Le moment normalisé d'ordre 2 permet ainsi d'établir une hiérarchie entre réseaux de points de différentes dimensions. Par exemple, au sens de $G(\Lambda)$, le réseau régulier optimal est respectivement A_1 A_2, A_3^*, D_4, D_5^*, E_6^*, E_7^*, E_8, D_{10}^+ pour les dimensions de 1 à 8 et 10 [35, 34]. Le réseau optimal est généralement le dual de l'empilement de sphères le plus dense dans chaque dimension [14], sans que ce soit toujours le cas [34]. Les réseaux optimaux ne sont pas forcément associés à des empilements de sphères [34].

Choix à bas débit : En pratique en codage bas débit, la distorsion moyenne ne se réduit pas à l'erreur granulaire. Elle dépend également de l'erreur de surcharge. De plus, l'erreur granulaire dépend cette fois-ci non seulement de la géométrie de $V(\Lambda)$ mais aussi de la distribution de la source. La hiérarchie des réseaux en terme de $G(\Lambda)$ n'est donc pas forcément valable. Par exemple, une source gaussienne généralisée sans mémoire a pour distribution de probabilité :

$$p(x) = \frac{A(\alpha)}{\sigma} e^{-|B(\alpha)\frac{x}{\sigma}|^\alpha}, \tag{2.35}$$

où $B(\alpha) = \sqrt{\frac{\Gamma(3/\alpha)}{\Gamma(1/\alpha)}}$, $A(\alpha) = \frac{2B(\alpha)}{2\Gamma(1/\alpha)}$, $\alpha > 0$ est le paramètre de décroissance exponentielle et σ^2 la variance ; pour $\alpha = 1$ et 2, on retrouve respectivement les lois laplacienne et gaussienne. Si $\alpha < 1$, le réseau $\mathbb{Z}$ peut être supérieur à E_8 et Λ_{24} [36].

Par ailleurs, si la quantification est effectuée suivant l'erreur quadratique, l'erreur maximale (norme l_∞) diffère suivant les réseaux.

2.3 Recherche du plus proche voisin dans un réseau infini (ou décodage de réseaux de points)

Le décodage d'un réseau de points Λ consiste à trouver le plus proche voisin d'un point $\mathbf{x}$ quelconque de $\mathbb{R}^n$ dans le réseau Λ infini. On se restreint au cas de la distance euclidienne. Pour $\mathbf{x} \in \mathbb{R}^N$, on cherche donc

$$\mathbf{y} = \arg\min_{\mathbf{c}\in\Lambda} ||\mathbf{x} - \mathbf{c}||^2. \tag{2.36}$$

Ce problème de recherche du plus proche voisin est également appelé décodage [37]. Cette terminologie provient du domaine des communications numériques sur canal perturbé par un bruit blanc gaussien additif. En effet, les réseaux de points sont utilisés en modulation codée en bloc pour définir des constellations multi-dimensionnelles (en général une séquence de constellations QAM) [37]. Dans un tel modem, la constellation est numérotée par des indices binaires. Le modulateur (codeur) se contente de former un indice à partir de bits d'information, d'associer à cet indice un point de la constellations et de générer les formes d'onde associées. Le démodulateur (décodeur) reçoit un point de la constellation entâché de bruit. Dans ce cas, la démodulation suivant le principe du maximum de vraisemblance consiste alors à trouver le plus proche voisin du point reçu dans la constellation. Les opérations de codage et décodage en quantification sont les duales de celles de la modulation codée, comme montrées à la figure 2.10.

Le développement d'algorithmes efficaces de décodage de réseau est en fait un problème en soi.

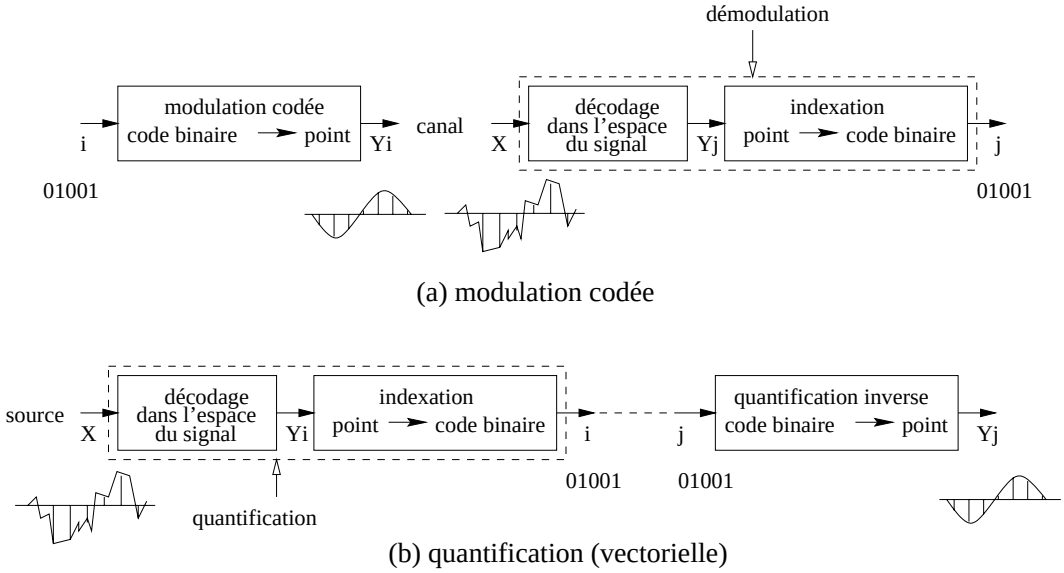


Figure 2.10: Dualité entre codage de source et de canal (quantification vectorielle et modulation codée).

On peut distinguer les algorithmes spécifiques à chaque réseau et les algorithmes génériques qui prennent comme paramètre la matrice génératrice $M(\Lambda)$ de Λ.

2.3.1 Algorithmes spécifiques

Cette recherche est généralement efficace en exploitant la structure spécifique de chaque réseau régulier. Les algorithmes rapides pour l'erreur quadratique ont d'abord été introduits par Conway et Sloane dans [13] pour les réseaux $\mathbb{Z}^N$, A_N, D_N, E_6, E_7 et RE_8 et dans [38] pour E_8 et Λ_{24}. Des algorithmes rapides de décodage pour les réseaux usuels sont énumérés dans le tableau 2.2. On détaille à l'annexe 2.D des algorithmes rapides de décodage pour $\mathbb{Z}^N$, D_N, RE_8 et Λ_{16}.

TABLEAU 2.2: Algorithmes de décodage des réseaux de points usuels.

Λ	Algorithme de décodage
A_2	Décodage par cosets [39]
	Décodage dans le plan $x+y+z=0$ [23]
A_N	Décodage dans l'hyperplan $\sum x_i = 0$ [13] [2, p. 447]
D_N	Décodage de $\mathbb{Z}^N$ avec règle de Wagner [13]
E_8	Viterbi sur le treillis de la "quadrature" [30]
	Décodage par le code de Hamming (8,4,4) [15]
RE_8	Viterbi sur le treillis de la "quadrature" [30]
	Décodage par cosets de $2D_8$[13]
	Décodage par points A_{ij}[37]
$R\Lambda_{16}$	Viterbi sur le treillis de la "double quadrature" [29]
Λ_{16}	Décodage par cosets de $2D_{16}$
	Viterbi sur le treillis de la "double quadrature" [29]
	Décodage par points A_{ij}[37]
	Décodage par le quadracode [40]
Λ_{24}	Viterbi sur le treillis de la "cubature" [29]
$R\Lambda_{24}$	Décodage par cosets de $2D_{24}$
	Décodage par points A_{ijk} et B_{ijk} [37]
	Viterbi sur le treillis de la "cubature" [30]
	Décodage par l'hexacode [41]

2.3.2 Algorithmes universels

Un algorithme universel de décodage réalise un décodage de Λ à partir d'une matrice génératrice $M(\Lambda)$ de Λ. Cette souplesse vient au prix d'une complexité non minimale (pour un réseau Λ donné et par rapport à un décodage spécifique).

Algorithme (quasi-universel) de Sayood-Blankenau

L'algorithme de [42] fonctionne en 2 étapes. Un point $\mathbf{v}$ de Λ proche de $\mathbf{x}$ est d'abord trouvé par arrondi des coordonnées de $\mathbf{x}$ dans la base de Λ. Ensuite, le plus proche voisin de $\mathbf{x}$ dans Λ est cherché autour de $\mathbf{v}$, sous la forme $\mathbf{v}+\delta$ où δ est pris dans un sous-ensemble approprié de Λ – un tel sous-ensemble peut être constitué des points de Λ autour de l'origine de norme inférieure à r. Le

point $\mathbf{v}$, appelé *point de Babai* [22], est défini par

$$\mathbf{v} = \mathbf{u}M(\Lambda) \tag{2.37}$$

avec

$$\mathbf{u} = [\mathbf{x}M(\Lambda)^{-1}], \tag{2.38}$$

où [.] désigne l'arrondi à l'entier le plus proche (par composante). Cet algorithme de décodage n'est pas tout à fait général, car il implique d'énumérer l'ensemble $\{\delta\}$ pour chaque Λ spécifique. Cette énumération n'est pas automatique.

Algorithme de Kannan

Le décodeur de Kannan étudié et modifié dans [43] consiste à chercher le plus proche voisin de $\mathbf{x}$ parmi les points de Λ contenus dans un parallépipède centré en $\mathbf{x}$, à partir d'une base réduite de Λ (idéalement réduite au sens de Korkine-Zolotareff).

Algorithme de Viterbo-Boutros basé sur la méthode de Pohst

Le *sphere decoder* de [44, 45] recherche le plus proche voisin de $\mathbf{x}$ parmi tous les points du réseau à l'intérieur d'une sphère centrée sur $\mathbf{x}$, le rayon de la sphère étant un paramètre de l'algorithme. Cette recherche repose entièrement sur l'algorithme de Pohst [46] de recherche du vecteur le plus court dans un réseau. En effet, minimiser $\|\mathbf{x}-\mathbf{c}\|^2$ pour $\mathbf{c} \in \Lambda$ est équivalent à minimiser $\|\mathbf{w}\|^2$ pour $\mathbf{w} \in \mathbf{c} - \Lambda$.

Le fonctionnement de l'algorithme est illustré à la figure 2.11 pour $\Lambda = D_2$, $\mathbf{x} = (3.1, 1.4)$ et une sphère de rayon $\sqrt{11}$. Le plus proche voisin de $\mathbf{x}$ dans D_2 est $\mathbf{y} = (3, 1)$. Il est trouvé après 5 itérations. Les 5 candidats successifs pour $\mathbf{y}$ dans D_2 sont numérotés dans l'ordre suivant lequel ils sont calculés et testés par l'algorithme.

L'énumération des points de Λ dans la sphère est effectuée suivant [46] en bornant l'intervalle

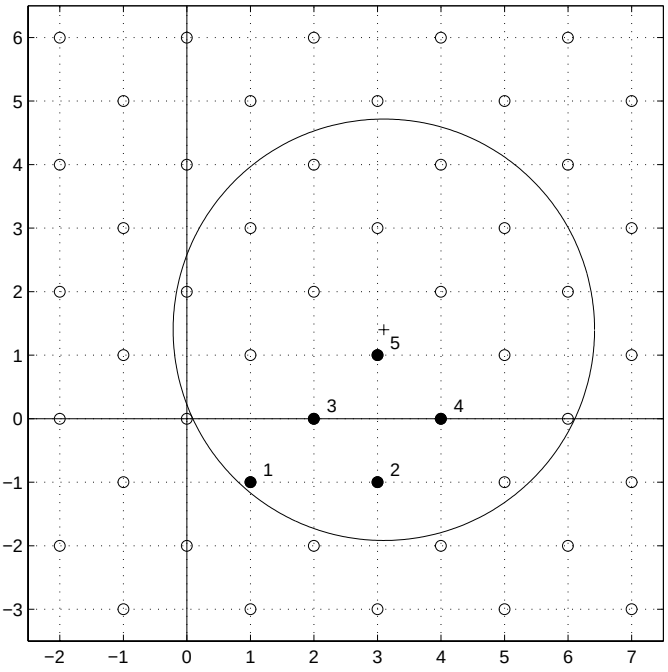


Figure 2.11: Exemple de recherche par l'algorithme de Viterbo-Boutros.

de recherche pour chaque composante. Pour décoder Λ, le rayon de la sphère peut être initialisé au rayon de couverture de Λ, et mis à jour pendant le déroulement de l'algorithme.

Algorithme de Schnorr-Euschner

L'algorithme de Schnorr-Euschner applique la méthode de Pohst de recherche du plus proche voisin dans une sphère, comme le *sphere decoder* de Viterbo-Boutros. Cependant, les candidats successifs calculés et testés par l'algorithme sont générés dans un ordre différent par rapport à la méthode de Pohst. La complexité de la recherche du plus proche voisin est alors réduite.

Le décodage de Schnorr-Euschner est réalisé en 2 étapes dans [22] : le plus proche voisin de $\mathbf{x}$ est d'abord estimé au point de Babai $\mathbf{v}$ associé à $\mathbf{x}$ dans Λ, puis les coordonnées de $\mathbf{v}$ dans Λ, $\mathbf{u}$, sont modifiées jusqu'à trouver le plus proche voisin. Le point de Babai est défini aux équations 2.37

et 2.38. Il dépend de $\mathbf{x}$, de Λ et aussi de la base utilisée pour représenter Λ (c'est-à-dire de $M(\Lambda)$). Ce n'est pas nécessairement le plus proche voisin de $\mathbf{x}$, mais l'erreur associée peut être majorée. La recherche du plus proche voisin est accélérée en utilisant une base réduite de Λ.

2.4 Techniques de quantification par réseau de points

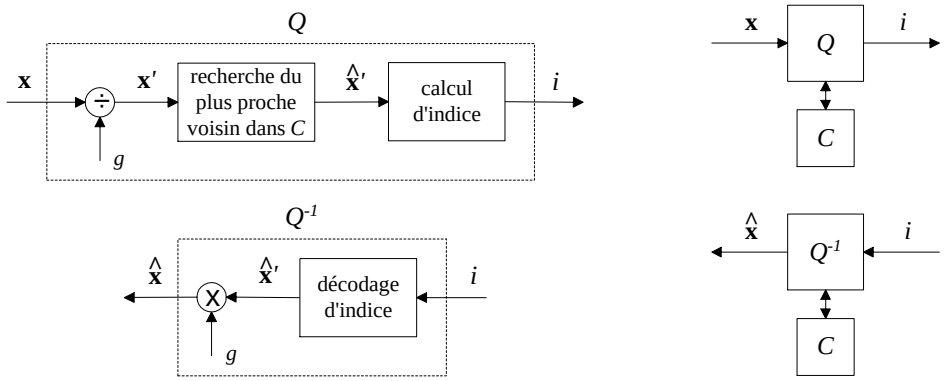
Les techniques de quantification par réseau de points sont revues ici en distinguant la quantification uniforme à débit fixe ou variable et la quantification non-uniforme à débit fixe.

2.4.1 Quantification uniforme à débit fixe

On rappelle qu'un réseau de points Λ contient un nombre infini de points. La quantification uniforme par Λ à débit fixe utilise comme dictionnaire C un sous-ensemble fini de Λ. En général, ce dictionnaire est construit en ne retenant qu'une orbite de Λ ou en tronquant Λ par une région bornée. On distingue donc deux types de dictionnaire : les dictionnaires définis par orbite et les dictionnaires obtenus par troncature.

Pour un dictionnaire C donné, la quantification est réalisée telle qu'à la figure 2.12 (a). Pour comparaison, les opérations équivalentes en quantification vectorielle non structurée sont rappelées à la figure 2.12 (b). En quantification par réseau de points (à débit fixe), le dictionnaire C n'est pas stocké et le codeur n'examine pas de façon exhaustive les mots de code (un par un). Par contre l'indice n'est pas calculé implicitement pendant la recherche du plus proche voisin.

La mise en œuvre de la quantification uniforme à débit requiert deux algorithmes : la recherche du plus proche voisin dans le dictionnaire C, qui est en général plus difficile qu'un décodage de Λ, et l'indexation (c'est-à-dire le calcul et le décodage de l'indice). Ces algorithmes dépendent en fait du réseau de points Λ utilisé et du type de dictionnaire choisi. Le problème de l'optimisation et de la quantification du facteur d'échelle g n'est pas étudié ici.



(a) quantification par réseau de points (b) quantification non structurée

Figure 2.12: Différences entre quantification non structurée et quantification par réseau de points.

Quantification par orbite

L'orbite d'un réseau de points Λ comprend tous les points $\mathbf{x} = (x_1, \cdots, x_N)$ de Λ situés à une certaine distance de l'origine (ou un point tel qu'un trou profond de Λ). La distance est généralement définie au moyen d'une norme l_p :

$$\|\mathbf{x} - \mathbf{y}\|_p = \left(\sum_{i=1}^{N} |x_i - y_i|^p \right)^{1/p} \tag{2.39}$$

si bien que les points d'une orbite obéissent à une équation de la forme [47, 19, 28]

$$\sum_{i=1}^{N} |x_i|^p = K^p, \tag{2.40}$$

où K est un rayon donné (pour la norme l_p). D'autres normes peuvent être également utilisées pour définir les orbites – par exemple, des normes l_1 ou l_2 pondérées [48].

La quantification par orbite d'un réseau de points a été introduite par Adoul et Fischer dans le contexte du codage de parole et d'images à bas débit [47, 19]. Elle est de type gain-forme (au sens de [49]) et permet d'utiliser des dictionnaires de taille réduite, en plus d'exploiter la redondance temporelle et la sensibilité logarithmique de la norme [47]. La forme des orbites doit idéalement correspondre à la forme des contours d'équiprobabilité de la source [50, 48] : elle est sphérique pour une source gaussienne (sans mémoire), pyramidale pour une source laplacienne (sans mémoire), etc.

On ne revoit ici que le cas des orbites sphériques et pyramidales.

Quantification sphérique :

La quantification sphérique utilise des orbites de la forme :

$$\sum_{i=1}^{N} x_i^2 = K^2. \tag{2.41}$$

Elle est adaptée à la source gaussienne sans mémoire. D'après [50], la quantification sphérique uniforme est optimale en dimension élevée – en accord avec la propriété d'équi-répartition asymptotique qui stipule que cette source est localisée en dimension élevée dans une mince coquille sphérique et que la distribution de la source dans cette coquille est uniforme (*typical set*) [4]. Néanmoins, en pratique, la dimension doit généralement être très élevée pour que la source soit réellement localisée sur une sphère [51].

Le recherche du plus proche voisin sur une orbite sphérique de Λ pour l'erreur quadratique revient à maximiser le produit scalaire entre le vecteur à quantifier et les points de l'orbite. Des algorithmes efficaces exploitant la structure (symétries, etc.) des orbites sont détaillées dans [52, 53] pour les réseaux RE_8 et Λ_{16} et dans [54] pour $R\Lambda_{24}$. Ces algorithmes sont adaptés au codage de l'excitation d'un filtre prédictif dans [18, 52].

La numérotation (ou indexation) requiert de disposer d'une énumération des différentes couches sphériques. Les orbites d'un réseau Λ peuvent être vues comme un ensemble de codes à permutations – avec des contraintes spécifiques à chaque Λ [53, 54]. La numérotation d'une orbite peut être effectuée sous la forme [53, 54, 55, 56, 57] : indice = classe + rang, où la classe identifie un code à permutation et le rang est l'indice d'une permutation. Dans le cas de $\mathbb{Z}^N$, l'indexation peut être totalement algorithmique [27]. Pour des réseaux tels que D_N, RE_8, Λ_{16} ou $R\Lambda_{24}$, elle peut être réalisée efficacement par le biais de vecteurs directeurs (ou leaders) ; l'indexation est alors en grande partie algorithmique, mais elle utilise également des tables. Le concept de vecteur directeur est détaillé à l'annexe 2.E dans le cas du réseau RE_8.

Les codes sphériques par réseau de points ont été étudiés dans [58] – des constructions (efficaces à débit élevé) de codes sphériques y sont également présentées ; celles-ci n'utilisent pas directement les orbites sphériques d'un réseau.

Quantification pyramidale :

La quantification pyramidale [19, 59] utilise des orbites de la forme

$$\sum_{i=1}^{N} |x_i| = K. \tag{2.42}$$

Cette technique est adaptée à la quantification d'une source laplacienne sans mémoire. Elle est conceptuellement très similaire à la quantification sphérique. Dans le cas du réseau cubique $\mathbb{Z}^N$, l'énumération des points sur chaque pyramide est récursive et l'indexation purement algorithmique [19]. L'indexation par leader pour des réseaux tels que D_N a été généralisée dans [55]. Une autre technique se trouve dans [60].

Quantification par réseau tronqué

Un exemple de troncature est donné à la figure 2.13 dans le cas du réseau A_2. Le dictionnaire C est défini en ne retenant que les points de A_2 qui sont à l'intérieur de la sphère de troncature. On peut vérifier que C a une forme sphérique. En fait, la troncature de réseau est également appelée mise en forme de réseau (*lattice shaping*).

La quantification par réseau tronqué généralise la quantification scalaire linéaire (utilisant le réseau $\mathbb{Z}$) à des dimensions $N \geq 1$. Pour optimiser ses performances pour une source stationnaire, la forme de la troncature doit idéalement être adaptée à la forme de la région de plus forte densité de probabilité de la source en dimension asymptotiquement élevée et aux contours d'équiprobabilité de la source [1]. La troncature est ainsi sphérique pour la source gaussienne sans mémoire, ellipsoïdale pour la source gaussienne vectorielle corrélée, pyramidale pour la source laplacienne sans mémoire, etc. Dans certains cas, la troncature est choisie pour simplifier la mise en œuvre – c'est le cas de la

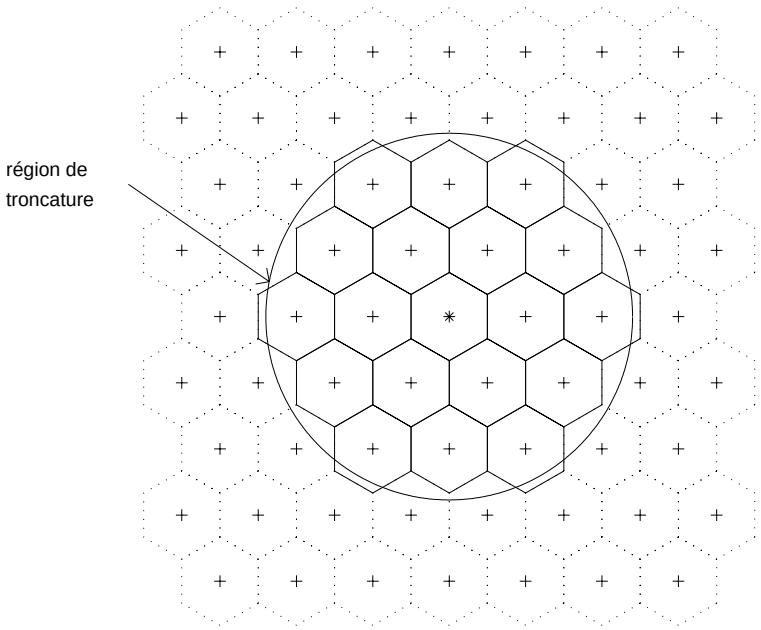


Figure 2.13: Exemple de troncature ($\Lambda = A_2$) : les croix représentent des points de Λ ; les points à l'intérieur de la région de troncature forment un dictionnaire de quantification C (avant mise à l'échelle).

quantification de Voronoï [15], ou pour s'adapter à l'espace de la source – c'est le cas de la troncature cubique adaptée à la quantification de phases de distribution uniforme dans $[0, \pi]^N$.

La recherche du plus proche voisin et l'indexation sont décrites succintement ci-dessous suivant la nature de la troncature :

- Cube : la recherche et l'indexation sont algorithmiques en utilisant le décodage par treillis de [30].

- Sphère, pyramide, ellipsoïde [61, 62, 63, 64, 65] : l'indexation est tirée des techniques de quantification par orbite. La recherche est en général effectuée en 2 étapes : décodage de Λ, puis saturation éventuelle par projection [63] ou réduction itérative [66], par recherche par orbite sphérique de rayon élevé [67], etc.

- Région de Voronoï d'un réseau Λ' géométriquement similaire à Λ : l'indexation (algorithmique) est décrite dans [15, 68]. La saturation dans un code de Voronoï est un problème peu étudié [69].

D'autres troncatures et techniques d'indexation sont définies par exemple dans [37, 70, 71].

2.4.2 Quantification uniforme à débit variable avec codage entropique

La quantification par réseau de points n'est optimale que pour une source uniforme. Dans le cas d'une source non-uniforme, la densité uniforme d'un réseau de points est inadaptée et pour augmenter l'efficacité du codage un codage entropique peut être appliqué après quantification.

On peut prendre deux approches pour le codage à débit variable des sorties d'un quantificateur : l'une séquentielle, l'autre conjointe. On peut ainsi optimiser un quantificateur pour la source et simplement représenter sa sortie par codage entropique (*entropy-coded quantization*) ou tenir compte du couplage entre quantification et codage entropique en introduisant une contrainte d'entropie dans le critère de distorsion (*entropy-constrained quantization*) [1, 72]. Ces deux approches sont examinées ici dans le cas de la quantification par réseau de points.

Quantification suivie par un codage entropique

L'entropie d'un quantificateur basé sur un dictionnaire C est

$$H = -\sum_{\mathbf{y}\in C} p(\mathbf{y}) \log_2 p(\mathbf{y}), \tag{2.43}$$

où $p(\mathbf{y})$ est la probabilité de sélection du mot de code $\mathbf{y}$. D'après la théorie de Shannon, H donne le débit minimal (en bits par vecteur) de codage de la sortie du quantificateur. Le codage entropique s'approche de cette limite inférieure en associant à chaque $\mathbf{y}$ un code binaire de longueur proche de $-\log_2 p(\mathbf{y})$ bits.

L'application directe du codage entropique au dictionnaire C (algébrique ou non) n'est envisageable qu'à faible dimension et pour des débits limités (en général ≤ 1 bit par dimension) [73]. Elle pose en effet des problèmes importants, tels que l'obtention de statistiques fiables et un stockage prohibitif des codes de longueur variable. Dans le cas de la quantification par réseau de points, une technique plus adaptée consiste à coder séparement l'orbite de $\mathbf{y}$ (préfixe) et l'indice de $\mathbf{y}$ sur cette orbite (suffixe) [74, 51]. Le préfixe est de longueur proche de $-\log_2 p(K)$, où K est le numéro de l'orbite de $\mathbf{y}$ et $p(K)$ est la probabilité de sélection de l'orbite K ; la longueur du suffixe est déterminée par le nombre de points N_K sur l'orbite en question, soit $\lceil \log_2 N_K \rceil$ bits. Cette technique est optimale si $p(\mathbf{y})$ est fonction de l'orbite de $\mathbf{y}$ et uniforme suivant l'angle de $\mathbf{y}$ [74]. Cette hypothèse est raisonnable dans le cas d'une source i.i.d. où les orbites correspondent à des contours d'équiprobabilité de la source. Cette technique est utilisée dans [27] pour une quantification dans $\mathbb{Z}^N$. Elle a été généralisée dans [56] où le codage de $\mathbf{y}$ est réalisé en codant séparément à débit variable la norme, le vecteur directeur (ou leader) et le rang de la permutation du vecteur directeur. On trouve dans [75] une comparaison de performances pour une quantification dans D_{10}^+.

Plutôt que d'appliquer un codage entropique par orbite, on peut coder les paramètres de troncature à longueur variable [76]. En quantification multi-débit les numéros de dictionnaire peuvent être codés à débit variable. Dans [77], le codage de Voronoï multi-débit dans $\Lambda/m\Lambda$ est appliqué avec un codage par plages (*run-length coding*) et de Huffman de m. Dans [78], le codage arithmétique de numéros de dictionnaire en quantification multi-débit est exploré.

Une technique totalement différente a été introduite dans [79], où la distribution de probabilité de la source est modélisée par un mélange de lois gaussiennes et un codage arithmétique est appliqué après quantification par $\mathbb{Z}^N$.

Quantification avec contrainte d'entropie

En quantification avec contrainte d'entropie, la quantification est optimisée pour un critère d'erreur de la forme

$$E[\|\mathbf{x} - Q(\mathbf{x})\|^2] + \lambda E[l(Q(\mathbf{x}))] \tag{2.44}$$

où $l(Q(x))$ désigne la longueur du mot de code $Q(\mathbf{x})$ et λ est un multiplicateur de Lagrange [80]. Généralement, ce critère est minimisé sans mesures réelles de $\|\mathbf{x} - Q(\mathbf{x})\|^2$ et $l(Q(\mathbf{x}))$ en estimant ces quantités avec une hypothèse sur la distribution de la source. Des modèles (ou estimations) de distorsion et d'entropie pour une quantification par réseau de points avec un codage séparé norme-position se trouvent dans [81, 82, 83, 84]. Ces modèles sont fonction du facteur d'échelle appliqué au réseau. Des techniques de quantification par réseau de points avec contrainte d'entropie sont présentées dans [85, 83, 86, 87].

2.4.3 Quantification non-uniforme à débit fixe

Plutôt que d'appliquer un codage entropique après quantification uniforme, la densité d'un réseau de points peut être adaptée à une source non-uniforme par non-linéarité sur la source (*companding*) ou par transformation de réseau.

La quantification scalaire uniforme peut être adaptée à l'aide d'une compression. Gersho a souligné la difficulté de définir une fonction de compression-expansion multi-dimensionnelle [7]. Dans [88], Moo et Neuhoff ont montré que pour une classe spécifique de quantificateurs, la fonction de compression multi-dimensionelle optimale est scalaire. Dans [11], une fonction de compression est développée à partir d'une union quasi-aléatoire de cosets de $\mathbb{Z}^N$.

Afin d'obtenir une densité de points non-uniforme, Jeong et Gibson ont introduit une technique de "dilatation" de réseau de points [63]. Dans [67], le réseau de points RE_8 a été transformé pour une source gaussienne en appliquant un facteur d'échelle optimal par orbite ou par vecteur directeur (ou leader) absolu.

2.5 Analyse asymptotique des performances de quantification

On reprend ici des résultats classiques de la théorie du codage asymptotique de Bennett, où la dimension N est fixée et le débit par dimension $R \to \infty$. La source est supposée stationnaire sans

mémoire.

La quantification par orbite a été analysée dans [19] pour des dimensions et des débits asymptotiques. On se concentre ici sur la quantification par réseau tronqué.

2.5.1 Quantification à débit fixe

On considère le cas d'une source $\mathbf{x}$ en dimension N. On suppose qu'un réseau de points Λ a été sélectionné, qu'on dispose d'un dictionnaire de quantification C obtenu par troncature de Λ et que la quantification est optimale, c'est-à-dire que tout vecteur $\mathbf{x}$ de la source est arrondi à son plus proche voisin $\hat{\mathbf{x}}$ dans un sous-ensemble de Λ.

La distorsion moyenne de quantification au sens de l'erreur quadratique, définie comme étant :

$$\bar{d}(R) = E\left[\frac{1}{N}||\mathbf{x}-\widehat{\mathbf{x}}||^2\right] = \frac{1}{N}\int_{\mathbb{R}^N} ||\mathbf{x}-\widehat{\mathbf{x}}||^2\, p(\mathbf{x})\, d\mathbf{x}, \tag{2.45}$$

où $p(\mathbf{x})$ est la distribution de probabilité de la source, peut se décomposer en distorsion granulaire $\bar{d}_g(R)$ et distorsion de saturation $\bar{d}_s(R)$. Plus précisément, $\bar{d}(R) = \bar{d}_g(R) + \bar{d}_s(R)$ avec :

$$\begin{cases} \bar{d}_g(R) = \frac{1}{N}\sum\limits_{\widehat{\mathbf{x}}\in C}\int_{V(\Lambda)+\widehat{\mathbf{x}}} ||\mathbf{x}-\widehat{\mathbf{x}}||^2\, p(\mathbf{x})\, d\mathbf{x} \\ \bar{d}_s(R) = \frac{1}{N}\int_{\mathbb{R}^N\setminus\{V(\Lambda)+C\}} ||\mathbf{x}-\widehat{\mathbf{x}}||^2\, p(\mathbf{x})\, d\mathbf{x} \end{cases} \tag{2.46}$$

où $V(\Lambda)$ est la région de Voronoï de Λ centrée autour de $\mathbf{0}$. La frontière de la région $\{V(\Lambda)+C\}$ correspond à la saturation effective de Λ.

Approximation de la distorsion granulaire

Dans l'hypothèse des débits élevés ($R \to \infty$), on arrive à une approximation analytique de la forme :

$$\bar{d}_g(R) \approx \mathrm{Vol}(V(\Lambda))^{2/N} G(\Lambda)(1-P_s) \tag{2.47}$$

où $\mathrm{Vol}(V(\Lambda))$ représente le volume de $V(\Lambda)$, $G(\Lambda)$ le moment normalisé d'ordre 2 de Λ et P_s la probabilité de saturation :

$$P_s = P\left[\mathbf{x} \notin \{V(\Lambda) + C\}\right] = 1 - \int_{\{V(\Lambda)+C\}} p(\mathbf{x})\, d\mathbf{x} \tag{2.48}$$

Le gain granulaire $\gamma_g(\Lambda)$, défini par

$$\gamma_g(\Lambda) = G(\mathbb{Z})/G(\Lambda) \tag{2.49}$$

est donc une caractéristique de Λ en quantification. Ce gain peut être majoré à l'aide de la la limite de la N-sphère [2], définie à l'équation 2.32. Pour une dimension de quantification fixée N, on choisit (si possible) le réseau Λ de plus petit moment normalisé d'ordre 2 – autrement dit, celui de meilleure granularité pour une source uniforme dans $V(\Lambda)$.

Approximation de la distorsion de saturation

Il est aussi possible de dériver une approximation analytique de $\bar{d}_s(R)$. On trouve de telles approximations dans [89, 69] pour les codes de Voronoï et dans [90] dans le cas général. En général ces modèles sont peu fiables pour des débits ≤ 3 bits par dimension.

2.5.2 Quantification à débit variable

On considère ici une source stationnaire sans mémoire de distribution de probabilité p et d'entropie différentielle $h(p) < \infty$. L'entropie différentielle $h(p)$ est définie par [4] :

$$h(p) = -\int_{-\infty}^{+\infty} p(x) log(p(x))\, dx. \tag{2.50}$$

La valeur de $h(p)$ pour des sources usuelles est rappelée au tableau 2.3 (d'après [91, p. 625]).

D'après Gish et Pierce [8], en quantification scalaire suivie d'un codage entropique, l'erreur quadratique moyenne $\bar{d}(R)$ pour un débit moyen de R bits par dimension est donnée par [8] :

$$\lim_{R\to+\infty} \bar{d}(R) 2^{2R} = \frac{2^{2h(p)}}{12}. \tag{2.51}$$

TABLEAU 2.3: Entropie différentielle pour quelques sources sans mémoire.

Source	Distribution p	Entropie différientielle $h(p)$
Gaussienne	$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}$	$\frac{1}{2}\log(2\pi e\sigma^2)$
Laplacienne	$p(x) = \frac{\lambda}{2} e^{-\lambda\|x\|}$ $(\lambda = \sqrt{\frac{2}{\sigma^2}})$	$\log(\frac{2e}{\lambda}) = \frac{1}{2}\log(2e^2\sigma^2)$
Gamma	$p(x) = \frac{\sqrt[4]{3}}{\sqrt{8\pi\sigma\|x\|}} e^{-\frac{\sqrt{3}\|x\|}{2\sigma}}$	$\frac{1}{2}\log_2(\frac{4\pi e^{1-C}}{3}\sigma^2) - C = 0.5772$ (constante d'Euler)

D'après Ziv [92], cette erreur devient dans le cas de la quantification par un réseau de points Λ suivie par un codage entropique :

$$\lim_{R\to+\infty} \bar{d}(R) 2^{2R} = G(\Lambda) 2^{2h(p)}. \tag{2.52}$$

où $G(\Lambda)$ est le moment normalisé d'ordre 2 de Λ. Ce résultat généralise le cas scalaire car $G(\mathbb{Z}) = 1/12$.

Or, d'après la théorie de la distorsion [93],

$$\lim_{R\to+\infty} \bar{d}(R) 2^{2R} = \frac{1}{2\pi e} 2^{2h(p)}. \tag{2.53}$$

De plus, il existe des réseaux réguliers tels que $G(\Lambda) \to \frac{1}{2\pi e}$ quand $N \to +\infty$. Par conséquent, lorsqu'elle est suivie par un codage entropique, la quantification par réseau régulier de points est en théorie plus performante que son équivalent scalaire (pour laquelle $G(\mathbb{Z}) = \frac{1}{12}$) et surtout – contrairement au cas scalaire – elle peut atteindre (asymptotiquement) la limite de Shannon. Elle permet donc de réduire l'écart de 1.53 dB (ou $\frac{1}{2}\log\frac{2\pi e}{12} \approx 0.255$ bit par dimension) entre la limite de Shannon et la quantification scalaire avec codage entropique. Cet avantage est dû à la réduction de l'erreur granulaire, où $G(\mathbb{Z}) = \frac{1}{12}$ est remplacé par $G(\Lambda)$.

2.6 Performances réelles et complexité

On dresse ici un aperçu du compromis performances-complexité en quantification par réseau de points par rapport à des techniques classiques de quantification. On se restreint à des sources sans

mémoire. Par conséquent, les seuls gains de la quantification vectorielle sur la quantification scalaire sont le gain granulaire γ_g (*granular gain*) et le gain d'entropie (*entropy gain*) qui se réduit au gain de forme γ_s (*shaping gain*) [94].

Pour éviter de décrire les conditions expérimentales de simulations, on se base essentiellement sur des résultats numériques et des mesures de complexité disponibles dans la littérature. Dans le cas de la quantification vectorielle non-structurée (LBG), des simulations ont néanmoins été conduites pour disposer de plus de résultats que dans [140]. Une base servant à la fois à l'apprentissage et à l'évaluation des performances, comprenant $20.10^6/N$ vecteurs de dimension N=2, 4, 8 ou 16, a été utilisée. Les dictionnaires ont étés optimisés par l'algorithme LBG [16] avec une initialisation aléatoire (en sélectionnant aléatoirement des vecteurs dans la base d'apprentissage).

2.6.1 Source gaussienne sans mémoire

La borne débit-distorsion $D(R)$ est donnée dans le cas gaussien par :

$$D(R) = \sigma^2 2^{-2R} \tag{2.54}$$

Les performances réelles de plusieurs techniques sont présentées dans le tableau 2.4 pour différentes dimension N et différents débits R par dimension. Cette comparaison inclut des techniques compétitives comme la quantification scalaire uniforme symétrique, la quantification scalaire non-uniforme (Lloyd-Max) [95, 96], la quantification vectorielle non structurée (LBG) [16], la quantification scalaire avec codage entropique [97], la quantification en treillis [98, 99] – scalaire ou vectorielle (TCQ ou TCVQ) – et la quantification scalaire-vectorielle (SVQ) issue de la quantification scalaire entropique [100]. En particulier, les codes en treillis (TCQ et TCVQ) s'appuient sur des structures régulières tout comme la quantification par réseau de points. Néanmoins, dans un cas la quantification est réalisée par une séquence de symboles, alors que dans l'autre la source est codée par blocs.

On prend ici comme référence de performances pour la quantification par réseau de points la quantification sphérique de [58], la quantification optimale dans RE_8 tronqué par une hypersphère [67], le codage de Voronoï dans Λ_{16} (sous-optimal) [69], la quantification dans $\mathbb{Z}^{16}$ tronqué par une

TABLEAU 2.4: Comparison de plusieurs techniques de quantification pour la source gaussienne sans mémoire – les valeurs sont données en terme de rapport signal sur bruit (en dB).

Technique	Débit R (en bit/dimension)					
	1	2	3	4	5	6
borne débit-distorsion	6.02	12.04	18.06	24.08	30.10	36.12
scalaire uniforme ($\mathbb{Z}$) [91]	4.40	9.25	14.27	19.38	24.57	29.83
Lloyd-Max [95, 97]	4.40	9.30	14.62	20.22	26.02	31.89
scalaire avec codage entropique [98, 97, 101]	4.64	10.55	16.56	22.55	28.57	34.59
LBG, $N = 2$ (simulations)	4.40	9.68	15.19	21.09		
LBG, $N = 4$ (simulations)	4.66	10.16				
LBG, $N = 8$ (simulations)	4.93					
TCQ à alphabet double [98] (256 états)	5.56	11.04	16.64			
TCVQ [99] (dimension 2, 16 états)	5.29	10.84	16.62	22.63		
SVQ [100] (dimension 4)	3.47	9.57	15.33			
SVQ [100] (dimension 32)	4.63	10.30	15.97			
quantification à 2 étages VQ-PLVQ [102]		10.42	16.26			
quantification à 2 étages VQ-(S)LVQ [102]		10.24	16.18			
quantification sphérique par $W_{\Lambda_{24}}$ [58]	2.44	11.02	17.36	23.33	29.29	35.27
troncature sphérique de RE_8 [67]		10.39				
troncature sphérique de $\mathbb{Z}^{16}$ avec code entropique [51]	4.56	10.45	16.38			
codage de Voronoï dans D_4 [69]		8.95	14.70	20.20	25.60	31.00
codage de Voronoï dans E_8 [69]		9.16	15.20	20.80	26.37	31.83
codage de Voronoï dans Λ_{16} [69]		9.27	15.50	21.40	26.94	32.62
$\mathbb{Z}^{16}$ uniforme [63]	3.99	10.07	15.52	21.00	26.16	32.07
$\mathbb{Z}^{16}$ uniforme par morceaux [63]	4.44	9.99	16.05	22.15	28.20	34.23

hypersphère (sous-optimale) [63] ou employant un codage entropique [51], et la quantification par dilatation de $\mathbb{Z}^{16}$ [63].

Performances à débit fixe

On peut d'abord observer que la quantification par réseau de points de [67, 69, 63] à débit fixe apporte un gain significatif par rapport à la quantification scalaire uniforme – surtout à un débit $\geq$ 2 bits par dimension. Cet avantage est dû au gain granulaire des réseaux RE_8 et Λ_{16} et surtout au gain de forme associé à une troncature sphérique [94, 89].

On peut également remarquer que la quantification par réseau de points peut concurrencer la quantification non structurée (Lloyd-Max ou LBG). En effet, en dimension 8 et 16 – avec RE_8 et Λ_{16}

– celle-ci est plus performante que la quantification scalaire de type Lloyd-Max et la quantification vectorielle non-structurée (LBG). On rappelle en outre que la quantification de type LBG n'est typiquement réalisable que pour $NR \leq 10$ bits.

Le codage par troncature sphérique de RE_8 [67] atteint un rapport signal-sur-bruit 0.1 dB en deça de la quantification scalaire entropique à 2 bit par dimension. Ce résultat doit néanmoins être nuancé par le fait que les performances de la quantification optimale dans RE_8 s'écartent de la borne débit-distorsion et des performances du scalaire entropique pour des débits ≥ 2 bits par dimension [67]. On retrouve ce même comportement dans le cas du codage de Voronoï évalué dans [69]. Ainsi, les dictionnaires obtenus par troncature (sphérique ou de Voronoï) perdent en efficacité par rapport au scalaire entropique ou à la borne débit-distorsion pour des débits ≥ 2 bits par dimension. Ce comportement est attribuable au fait qu'à débit élevé la densité uniforme des réseaux de points devient de plus en plus sous-optimale par rapport à la densité optimale, et donc l'entropie des dictionnaires uniformes diminue relativement au débit fixe des dictionnaires. Pour garantir des performances à débit fixe équivalentes ou supérieures au scalaire entropique sur une large gamme de débits, on doit plutôt recourir à :

- une quantification à débit fixe sur orbite sphérique en dimension élevée [47, 58] ;

- une quantification à débit fixe avec "dilatation" de réseau [63].

En comparaison, la quantification en treillis (TCQ et TCVQ) et la quantification scalaire-vectorielle (SVQ) atteignent de bonnes performances à bas débit. Néanmoins, dans le cas de la quantification en treillis (TCQ), cette performance s'écarte de la borne débit-distorsion à mesure que le débit augmente ; ce problème peut être corrigé en appliquant un codage entropique [103]. Le codage hybride stochastique-algébrique de [102] est également une technique performante.

Performances à débit variable

On peut tout d'abord vérifier que pour la source gaussienne, comme l'ont montré Goblick et Holsinger [104], la performance de la quantification scalaire avec codage entropique – ou simplement scalaire entropique – est à 1.53 dB (0.255 bit par dimension) de la borne débit-distorsion.

La quantification par troncature sphérique de $\mathbb{Z}^{16}$ avec codage entropique des numéros de sphères [51] atteint des performances très proches de celles de la quantification scalaire entropique. Ce résultat pourrait a priori être amélioré en employant un réseau possédant plus de structure que $\mathbb{Z}^{16}$, par exemple RE_8, Λ_{16} ou Λ_{24}. Néanmoins, dans ce cas, l'énumération générale des sphères peut devenir complexe et non-triviale [52, 54]. De plus, le codage entropique de la sortie d'un quantificateur par réseau de points doit s'adapter à des dictionnaires de taille qui grandit exponentiellement avec le débit ou la dimension.

2.6.2 Complexité

Par rapport à la quantification scalaire uniforme, le désavantage de la quantification vectorielle par réseau de points est l'augmentation de complexité. Même si des algorithmes rapides de décodage de réseaux existent [13], ceux-ci requièrent plus d'opérations que des simples arrondis. En quantification vectorielle par réseau de points, la surcharge est difficile à gérer et l'indexation explicite peut devenir très complexe. La quantification par réseau de points ne doit donc pas être tenue pour être la panacée, d'autant plus que sa conception et son optimisation pour une source donnée n'ont parfois rien de trivial.

Pour juger du compromis entre performances et complexité, la complexité de calcul de plusieurs techniques est donnée au tableau 2.5.

Pour des performances proches, les quantifications sphérique de [58] et en treillis de [98] ont des complexités similaires, linéaires en fonction du débit R. À l'opposé la complexité de la quantification vectorielle non structurée est exponentielle en fonction du débit et de la dimension.

TABLEAU 2.5: Comparaison des complexités pour la quantification d'une source gaussienne sans mémoire – N =dimension, R=débit (par dimension), S =nombre d'états du treillis.

Technique	Nombre de calculs (une opération = une fonction arithmétique [105])
quantification sphérique par $W_{\Lambda_{24}}$ [58]	$R + 126$
TCQ à alphabet double [98]	$3S + 4R + 4$
LBG [16]	2^{NR+1}

2.6.3 Source laplacienne sans mémoire

La borne débit-distorsion $D(R)$ n'est pas connue explicitement sous forme analytique pour des sources sans mémoire non-gaussiennes. Pour ces sources, il existe cependant des bornes sur $D(R)$ [91, pp. 640–641] et en particulier la limite inférieure de Shannon $D(R) \geq \frac{1}{2\pi e} 2^{-2R} 2^{h(p)}$. Cette limite inférieure converge vers la borne $D(R)$ à débit élevé ; à débit faible, la borne $D(R)$ doit être calculée numériquement par l'algorithme d'Arimoto-Blahut [106].

Les performances réelles de plusieurs techniques sont montrées au tableau 2.6. Comme dans le cas de la source gaussienne, plusieurs techniques compétitives sont comparées.

On peut observer d'abord comme Farvardin et Modestino [101] que pour la source laplacienne la quantification scalaire entropique est à 1.5 dB de la borne débit-distorsion, même à bas débit.

On peut faire les mêmes observations que dans le cas de la source gaussienne sans mémoire. La quantification par réseau de points est compétitive – par rapport à la quantification scalaire entropique, à la quantification en treillis (TCQ) ou la quantification scalaire-vectorielle (SVQ) – à condition d'utiliser le codage sur orbite pyramidale (PVQ) [19], le codage par troncature pyramidale avec "dilatation" de réseau [63] ou une quantification par réseau tronqué par une hyperpyramide avec codage entropique.

TABLEAU 2.6: Comparison de plusieurs techniques de quantification pour la source laplacienne sans mémoire – les valeurs sont données en terme de rapport signal sur bruit (en dB).

Technique	Débit (en bit/dimension)					
	1	2	3	4	5	6
borne débit-distorsion	6.62	12.66	18.68	24.69	30.73	36.75
scalaire uniforme ($\mathbb{Z}$) [91]	3.01	7.07	11.44	15.96	20.60	25.36
Lloyd-Max [95, 97]	3.01	7.54	12.64	18.13	23.87	29.74
scalaire avec codage entropique [98, 97, 101]	5.76	11.31	17.20	23.16	29.19	35.22
LBG, $N = 2$ (simulations)	3.66	8.84	14.29	20.06		
LBG, $N = 4$ (simulations)	4.63	9.99				
LBG, $N = 8$ (simulations)	5.28					
TCQ à alphabet double [98] (256 états)	4.83	9.90	15.37			
SVQ [100] (dimension 4)	3.78	9.38	13.99			
SVQ [100] (dimension 32)	5.59	10.87	16.14			
quantification à 2 étages VQ-PLVQ [102]		10.55	16.34			
quantification à 2 étages VQ-(S)LVQ [102]		10.37	16.09			
quantification dans A_4^* par quadrant [17]	3.97	9.81				
quantification pyramidale dans $\mathbb{Z}^{16}$ (PVQ) [19, 63]	5.09	10.42	16.33	22.40	28.68	34.35
quantification pyramidale dans $\mathbb{Z}^{32}$ (PVQ) [19]	5.30	11.09	16.77			
troncature pyramidale de $\mathbb{Z}^{16}$ [63]	4.81	10.38	15.45	20.65	25.82	31.13
$\mathbb{Z}^{16}$ uniforme par morceaux [63]	5.32	10.07	16.23	22.31	28.36	34.41

2.7 Conclusions et aspects connexes

Ce chapitre a présenté une approche de quantification uniforme, exploitant la structure régulière (linéaire) des réseaux de points : la quantification par réseau de points. On s'est restreint ici au cas où la quantification utilise un seul dictionnaire (fixe). Cette technique est élégante sur le plan conceptuel, asymptotiquement optimale (sous certaines conditions et avec un codage entropique). Surtout elle offre potentiellement un excellent compromis entre performances et complexité (nombre d'opérations et stockage) car il existe des algorithmes rapides de recherche du plus proche voisin dans un réseau de points (infini). Néanmoins, sur le plan pratique, elle n'est sans poser quelques problèmes importants. Le problème le plus ardu est celui de la mise en œuvre de dictionnaires de quantification C de taille finie à partir d'un réseau de points Λ (infini). Deux sortes d'algorithmes sont nécessaires : l'indexation des mots de code dans C et la recherche du plus proche voisin dans C. L'indexation est explicite (au codage Q comme au décodage Q^{-1}) et peut être complexe à débit

et dimension élevés, dès lors que des réseaux autres que $\mathbb{Z}^N$ sont utilisés. Le fait d'utiliser un sous-ensemble fini de Λ peut ralentir significativement la recherche du plus proche voisin par rapport à une recherche dans Λ (infini). Par ailleurs, les performances et complexité pour des sources artificielles sans mémoire (gaussienne et laplacienne) montrent que la quantification par réseau de points est compétitive à condition d'utiliser des techniques telles qu'une quantification par orbite de forme adaptée à la source [47, 19, 58], ou une quantification par troncature adaptée à la source avec codage entropique [74, 51, 27, 56, 75], ou encore une quantification par réseau "dilaté" [63].

Plusieurs aspects n'ont pas été revus, dont la quantification par réseau de points multi-débit [66], la robustesse de la quantification par réseau de points aux erreurs binaires [107, 108, 109] ou pertes de paquets [110, 111], ou la quantification par raffinements successifs [112]. La façon d'appliquer cette technique de quantification à des problèmes de codage de parole [18, 52, 113, 114, 102, 115, 67, 116, 117, 118, 119, 120], audio [121, 66, 122], d'images [123, 124, 25, 108, 109, 125, 73, 81, 126, 112, 69, 127, 128, 129, 130, 131, 132, 133] ou vidéo [70, 77, 134, 135, 136] n'a pas été présentée, pour ne pas rentrer dans des détails propres à chaque système de codage. Le couplage entre quantification par réseau de points et allocation adaptative de bits a par exemple été omis. Enfin, les performances de quantification pour des sources avec mémoire ou des sources réelles (coefficients de transformée, etc.) n'ont pas été revues.

Annexe 2.A : Matrices génératrices ou de Gram des réseaux importants

Les réseaux importants en dimension $N \geq 24$ sont spécifiés ici par matrices génératrices ou matrices de Gram. On rappelle que le réseau cubique $\mathbb{Z}^N$ est spécifié par la matrice génératrice identité.

Famille A_N $(N \geq 1)$: La matrice de Gram de A_N est tridiagonale :

$$A(A_N) = \begin{bmatrix} 2 & -1 & & \\ -1 & 2 & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{bmatrix} \tag{2.55}$$

– les coefficients non spécifiés sont nuls. A_2 est le réseau hexagonal. A_3 est le réseau cubique à faces centrées (structure cristalline du cuivre Cu, du chlorure de sodium NaCl, etc.).

Famille A_N^* $(N \geq 1)$: La matrice de Gram de A_N^* est :

$$A(A_N^*) = \begin{bmatrix} N & -1 & \cdots & -1 & -1 \\ -1 & N & \ddots & \vdots & -1 \\ \vdots & \ddots & \ddots & -1 & \vdots \\ -1 & \cdots & -1 & N & -1 \\ -1 & -1 & \cdots & -1 & N \end{bmatrix} \tag{2.56}$$

A_3^* est le réseau cubique centré (structure cristalline du fer).

Famille D_N $(N \geq 2)$: Une matrice génératrice de D_N est

$$M(D_N) = \begin{bmatrix} 2 & & & \\ 1 & 1 & & \\ \vdots & & \ddots & \\ 1 & & & 1 \end{bmatrix} \tag{2.57}$$

$D_2 \cong \mathbb{Z}^2$ est le réseau en damier (par analogie avec la table de jeu d'échecs). $D_3 \cong A_3$. D_4 est le réseau de Schläfli.

Famille D_N^* ($N \geq 2$) : Une matrice génératrice de D_N^* est

$$M(D_N^*) = \begin{bmatrix} 2 & & & & \\ & \ddots & & & \\ & & & 2 & \\ 1 & 1 & 1 & 1 \end{bmatrix} \tag{2.58}$$

$D_3^* \cong A_3^*$.

Famille D_N^+ (N pair ≥ 4): Une matrice génératrice de D_N^+ est

$$M(D_N^+) = \begin{bmatrix} 2 & & & & \\ 1 & 1 & & & \\ \vdots & & \ddots & & \\ 1 & & & 1 & \\ \frac{1}{2} & \frac{1}{2} & \cdots & \frac{1}{2} & \frac{1}{2} \end{bmatrix} \tag{2.59}$$

D_8^+ est le réseau de Gosset ($\frac{1}{2}RE_8$).

Famille E_N et E_N^* ($N = 6, 7, 8$) : La famille E_N n'est définie que pour $N = 6, 7, 8$ – de même que E_N^*. E_6 est un sous-réseau de A_7^* et E_7 est un sous-réseau de D_8^*. E_8 est équivalent à D_8^+ ; il est auto-dual, i.e. $E_8^* = E_8$. Les réseaux E_6, E_6^*, E_7 et E_7^* sont spécifiés dans [2].

Autres réseaux : Des matrices génératrices pour le réseau de Coxeter K_{12}, le réseau de Barnes-Wall Λ_{16} en dimension 16 et le réseau de Leech $R\Lambda_{24}$ se trouvent dans [2].

Annexe 2.B : Invariance de $G(\Lambda)$ à la dilatation

Le moment normalisé d'ordre 2, $G(\Lambda)$, d'un réseau Λ de $\mathbf{R}^N$ est défini à l'équation 2.31. Si le réseau Λ est dilaté d'un facteur $\alpha > 0$, on trouve :

$$G(\alpha\Lambda) = \frac{1}{N} \frac{1}{vol(V(\alpha\Lambda))^{1+2/N}} \int_{V(\alpha\Lambda)} \|\mathbf{x}\|^2 \, d\mathbf{x}. \tag{2.60}$$

Or, $V(\alpha\Lambda) = \alpha V(\Lambda)$ et $vol(V(\alpha\Lambda)) = \alpha^N vol(V(\Lambda))$. L'équation 2.60 peut donc être écrite sous la forme :

$$G(\alpha\Lambda) = \frac{1}{N} \frac{1}{\alpha^{N+2} vol(V(\Lambda))^{1+2/N}} \int_{\alpha V(\Lambda)} \|\mathbf{x}\|^2 \, d\mathbf{x}. \tag{2.61}$$

Le changement de variable $\mathbf{x} = \alpha\mathbf{y}$, avec $d\mathbf{x} = \alpha^N d\mathbf{y}$, donne :

$$\int_{\alpha V(\Lambda)} \|\mathbf{x}\|^2 \, d\mathbf{x} = \int_{V(\Lambda)} \alpha^2 \|\mathbf{y}\|^2 \, \alpha^N d\mathbf{y} = \alpha^{N+2} \int_{V(\Lambda)} \|\mathbf{y}\|^2 \, d\mathbf{y}. \tag{2.62}$$

Par suite, $G(\alpha\Lambda) = G(\Lambda)$. Le facteur $1/vol(V(\Lambda))^{2/N}$ rend donc le moment d'ordre 2 invariant à la dilatation.

Annexe 2.C : Estimation du moment normalisé d'ordre 2

L'intégrale

$$G(\Lambda) = \frac{1}{N}\frac{1}{Vol(V(\Lambda))^{1+2/N}}\int_{V(\Lambda)} \|\mathbf{x}\|^2 d\mathbf{x} \tag{2.63}$$

peut être estimée par la méthode de Monte Carlo à l'aide d'un générateur aléatoire uniforme. Le code MATLAB correspondant est donné ci-dessous dans le cas du réseau A_2 :

```
L = 1000000;                % nombre de tirages aléatoires
N=2;                        % dimension
M = [1 0; 1/2 sqrt(3)/2]; % matrice génératrice de A2

G=0.0;                      % moment d'ordre 2 normalisé
for i=1:L
  % tirage aléatoire d'une source uniforme dans V(A2)
  k=rand(1,N);
  u=k*M;       % u est dans le parallelotope de A2
  v=A2_dec(u); % recherche du plus proche voisin dans A2 (i.e. modulo A2)
  x=u-v;       % x est un point de V(A2)

  % accumulation de l'erreur quadratique
  G=G+sum(x.^2);
end
G = G/L/(N*abs(det(M))^(2/N)) % valeur de G(A2) pour Vol(V(A2))=1
```

Annexe 2.D : Algorithmes rapides de recherche du plus proche voisin (décodage) pour $\mathbb{Z}^N$, D_N, $2D_N^+$ et Λ_{16}

Décodage de $\mathbb{Z}^N$: Pour un scalaire x, on définit la fonction $[x]$ comme l'arrondi à l'entier le plus proche de x. Le décodage d'un point $\mathbf{x} = (x_1, \cdots, x_N) \in \mathbb{R}^N$ dans $\mathbb{Z}^N$ se réduit à calculer $\mathbf{y} = ([x_1], \cdots, [x_N])$.

Décodage de D_N ($N \geq 2$) : Le décodage dans D_N s'appuie sur le décodage dans $\mathbb{Z}^N$ et la règle de Wagner [137]. Ses étapes sont :

1. Calculer le plus proche voisin $\mathbf{y} = (y_1, \cdots, y_N)$ de $\mathbf{x}$ dans $\mathbb{Z}^N$, ainsi que $s = y_1 + \cdots + y_N$;
2. Si la somme s est impaire, appliquer la règle de Wagner à $\mathbf{y}$:
   (i) Localiser l'erreur d'arrondi maximale
   $$j = \arg \max_{1 \leq i \leq N} |x_i - y_i| \tag{2.64}$$
   (ii) Si $x_j - y_j < 0$, $y_j := y_j - 1$, sinon $y_j := y_j + 1$

En cas de violation de parité, la règle de Wagner consiste donc à arrondir une des composantes de $\mathbf{x}$ dans le mauvais sens ; pour minimiser l'erreur quadratique, on change la composante pour laquelle l'erreur d'arrondi est maximale.

Décodage de $2D_N^+$ (N pair ≥ 4) : Le réseau $2D_N^+$ est constitué de 2 cosets de $2D_N$, puisque par définition [2]

$$2D_N^+ = 2D_N \cup \{2D_N + [11 \cdots 1]\} \tag{2.65}$$

Le décodage de $2D_N^+$ peut donc s'appuyer sur le décodage par cosets de [13]. Ses étapes sont :

1. Calculer le plus proche voisin $\mathbf{y}_0$ de $\mathbf{x}$ dans $2D_N$

2. Calculer le plus proche voisin $\mathbf{y}_1$ de $\mathbf{x}$ dans $2D_N + [11\cdots 1]$:

   (i) Calculer $\mathbf{x}_1 = \mathbf{x} - [11\cdots 1]$

   (ii) Calculer le plus proche voisin $\mathbf{y}$ de $\mathbf{x}_1$ dans $2D_N$

   (iii) Calculer $\mathbf{y}_1 = \mathbf{y} + [11\cdots 1]$

3. Si $\|\mathbf{x} - \mathbf{y}_0\|^2 < \|\mathbf{x} - \mathbf{y}_1\|^2$, $\mathbf{y} = \mathbf{y}_0$ sinon $\mathbf{y} = \mathbf{y}_1$.

Cet algorithme utilise deux décodages dans $2D_N$. Ce dernier se déduit facilement du décodage dans D_N en faisant l'arrondi dans $2\mathbb{Z}^N$, puis en appliquant la règle de Wagner si la somme des composantes n'est pas un multiple de 4 et le cas échéant en modifiant la composante d'erreur maximale par ± 2.

Décodage de Λ_{16} : Le réseau Λ_{16} est constitué de 32 cosets de $2D_{16}$:

$$\Lambda_{16} = \bigcup_{\mathbf{c}\in(16,5,8)} 2D_{16} + \mathbf{c} \tag{2.66}$$

où le code binaire $(16, 5, 8)$ est le code de Reed-Muller d'ordre 1 de matrice génératrice

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{2.67}$$

Le décodage de Λ_{16} – comme celui de $2D_N^+$ – peut être réalisé par cosets d'après le principe de [13]. L'algorithme associé n'est pas détaillé par souci de concision ; sa complexité peut être réduite en éliminant des calculs redondants (arrondis et calculs de métriques).

Comparaison des complexités de décodage : La complexité de décodage pour les réseaux $\mathbb{Z}^N$, D_N, $2D_N^+$ et Λ_{16} est comparée au tableau 2.7. Le calcul de valeurs absolues et du signe ne sont pas comptabilisés. L'aglorithme de décodage par cosets de Λ_{16} est supposé non optimisé. Cette comparaison montre néanmoins que plus la structure du réseau est complexe plus le nombre d'opérations est élevé.

TABLEAU 2.7: Comparaisons des complexités (maximales) de décodage pour les réseaux $\mathbb{Z}^N$, D_N, $2D_N^+$ et $R\Lambda_{16}$.

Nombre d'opérations	$\mathbb{Z}^N$	D_N	$2D_N^+$	Λ_{16} $(N = 16)$
Additions		$2N$	$8N$	$158N$
Multiplications			$2N$	$32N$
Comparaisons		$N-1$	$2N-1$	$32N-1$
Arrondis	N	N	$2N$	$32N$

Annexe 2.E : Numérotation des orbites sphériques de RE_8 par leaders

Le réseau RE_8 est un réseau de points important en dimension 8 – relativement simple –, qui trouve plusieurs applications pratiques en quantification [52, 53, 66]. Il peut être défini de plusieurs façons. D'abord à partir de D_8 :

$$RE_8 = 2D_8^+ = 2D_8 \bigcup \{2D_8 + [11\cdots 1]\}\,. \tag{2.68}$$

Alternativement par la formule de [30] :

$$RE_8 = 4\mathbb{Z}^8 + 2(8,7,2) + (8,1,8) = \left\{\mathbf{x} = 4\mathbf{z} + 2\mathbf{c} + \mathbf{d} | \mathbf{z} \in \mathbb{Z}^8, \mathbf{c} \in (8,7,2), \mathbf{d} \in (8,1,8)\right\} \tag{2.69}$$

où les code binaires $(8,7,2)$ et $(8,1,8)$ sont respectivement le codes à parité (paire) et le code à répétition. Ou simplement en observant que RE_8 comprend tous les points $\mathbf{x} = (x_1, \cdots, x_8)$ tels que :

1. $\mathbf{x} \in \mathbb{Z}^8$, soit $\mathbf{x}$ est un vecteur entier ;
2. la somme $x_1 + x_2 + \cdots + x_8$ des composantes de $\mathbf{x}$ est un multiple de 4 ;
3. les composantes de $\mathbf{x}$ sont ou bien toutes paires ou bien toute impaires.

Les contraintes dans cette dernière définition se déduisent facilement de l'équation 2.69 – en particulier la contrainte 2 est due au code $(8,7,2)$ et la contrainte 3 au code $(8,1,8)$.

Le réseau RE_8 est intégral : ses orbites sphériques (sphères) ont un rayon $\sqrt{8m}$, où m est un entier ≥ 0. Cette propriété est facile à vérifier en développant $\|\mathbf{x}\|^2$ à partir de l'équation 2.69. On note $RE_8(m)$ la sphère de RE_8 de rayon $\sqrt{8m}$. Alors :

$$RE_8 = \bigcup_{m \geq 0} RE_8(m) \tag{2.70}$$

L'énumération des points des premières sphères $RE_8(m)$ a été réalisée dans [47, 52, 53] à l'aide du concept de vecteur directeur (ou leader).

On définit ainsi une classe d'équivalence sur RE_8 :

$$\mathbf{x} \approx \mathbf{y} \tag{2.71}$$

si et seulement si $\mathbf{y}$ est une permutation des composantes de $\mathbf{x}$. Par exemple, $(2,0,-2,0,0,0,0,0) \approx (0,0,0,0,2,-2,0,0)$. Cette relation induit une partition de RE_8 en classes d'équivalence qui sont des codes à permutations. En considérant l'ordre lexicographique, on définit comme *leader signé* le maximum de chaque classe d'équivalence. L'énumération de $RE_8(m)$ par leaders signés permet d'appliquer les algorithmes de recherche du plus proche voisin de [138] et d'indexer chaque point de $RE_8(m)$ grâce à un numéro de classe et un rang de permutation (calculé par exemple par la formule de [139]). Grâce au concept de leader signé, le codage d'un vecteur quelconque peut être effectué de manière optimale en comparant ce vecteur à quelques leaders uniquement. Cette recherche du plus proche voisin est en général beaucoup plus rapide qu'une recherche exhaustive dans un dictionnaire stochastique. Néanmoins, l'indexation est explicite, sa complexité algorithmique est non négligeable et elle requiert de stocker quelques tables d'indexation.

Pour réduire le nombre de leaders et accélérer la recherche du plus proche voisin, on peut définir un nouveau partitionnement de $RE_8(m)$:

$$\mathbf{x} \sim \mathbf{y} \tag{2.72}$$

si et seulement si les valeurs absolues des composantes de $\mathbf{x}$ et $\mathbf{y}$ arrangées dans l'ordre décroissant sont identiques. Par exemple, $(2,0,-2,0,0,0,0,0) \sim (0,0,2,0,2,0,0,0)$. Cette relation induit une nouvelle partition de RE_8 en classes d'équivalence qui peuvent aussi être interprétées comme des codes à permutations. En considérant l'ordre lexicographique, on définit comme *leader absolu* le maximum de chaque classe d'équivalence. Néanmoins, les permutations des leaders absolus sont cette fois-ci signées et le nombre de composantes négatives est contraint pour chaque leader absolu [52].

L'indexation par leader introduite par Adoul dans [18, 47] a été étendue à $R\Lambda_{24}$ dans [54] et à d'autres réseaux dans [55, 56, 57].

BIBLIOGRAPHIE

[1] A. Gersho and R.M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic Publishers, 1992.

[2] J.H. Conway and N.J.A. Sloane, *Sphere Packings, Lattices and Groups*, Springer-Verlag, 3rd edition, 1999.

[3] N.J.A. Sloane, "The packing of spheres," Scientific American, Jan. 1984.

[4] T. Cover and J. Thomas, *Elements of Information Theory*, Wiley & Sons, 1991.

[5] I.F. Blake, "The Leech Lattice as a Code for the Gaussian Channel," *Information and Control*, vol. 19, pp. 66–74, 1971.

[6] R. DeBuda, "The Upper Bound of a New Near-optimal Code," *IEEE Trans. Inform. Theory*, vol. 21, pp. 441–445, May 1975.

[7] A. Gersho, "Asymptotically optimal block quantization," *IEEE Trans. Inform. Theory*, vol. 25, pp. 373–380, 1979.

[8] H. Gish and J. N. Pierce, "Asymptotically Efficient Quantizing," *IEEE Trans. Inform. Theory*, vol. 14, pp. 678–683, Sep. 1968.

[9] R.C. Wood, "On optimum quantization," *IEEE Trans. Inform. Theory*, vol. 15, pp. 248–252, 1969.

[10] W.R. Bennett, "Spectra of quantized signals," *Bell Syst. Tech. J.*, vol. 27, pp. 446–472, July 1948.

[11] T.Z. Shabestary and P. Hedelin, "Spectral quantization by companding," in *Proc. ICASSP*, 2002, vol. I, pp. 641–644.

[12] N.J.A. Sloane, "Tables of Sphere Packings and Spherical Codes," *IEEE Trans. Inform. Theory*, vol. 27, no. 3, pp. 327–338, May 1981.

[13] J.H. Conway and N.J.A. Sloane, "Fast Quantizing and Decoding Algorithms for Lattice Quantizers and Codes," *IEEE Trans. Inform. Theory*, vol. 28, no. 2, pp. 227–231, Mar. 1982.

[14] J.H. Conway and N.J.A. Sloane, "Voronoi Regions of Lattices, Second Moments of Polytopes, and Quantization," *IEEE Trans. Inform. Theory*, vol. 28, no. 2, pp. 211–226, Mar. 1982.

[15] J.H. Conway and N.J.A. Sloane, "A fast encoding method for lattice codes and quantizers," *IEEE Trans. Inform. Theory*, vol. 29, pp. 820–824, Nov. 1983.

[16] Y.I. Linde, A. Buzo, and R.M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Commun.*, vol. 28, pp. 84–95, Jan. 1980.

[17] K. Sayood, J.D. Gibson, and M.C. Rost, "An algorithm for Uniform Vector Quantizer Design," *IEEE Trans. Inform. Theory*, vol. 30, no. 6, pp. 805–814, Nov. 1984.

[18] J.-P. Adoul, C. Lamblin, and A. Leguyader, "Baseband speech coding at 2400 bps using spherical vector quantization," in *Proc. ICASSP*, 1984, vol. 1, pp. 1.12.1–1.12.4.

[19] T.R. Fischer, "A pyramid vector quantizer," *IEEE Trans. Inform. Theory*, vol. 32, pp. 568–583, 1986.

[20] P. DeBuda, "Encoding and decoding algorithms for an optimal lattice-based code," in *Int. Conf. on Commun.*, Jun. 1981, pp. 65.3.1–65.3.5.

[21] J.H. Conway and N.J.A. Sloane, "Fast 4- and 8-dimensional quantizers and decoders," in *National Telecommunications Record, IEEE Press, NY*, 1981, vol. 3, pp. F4.2.1–F4.2.4.

[22] E. Agrell, T. Eriksson, A. Vardy, and K. Zeger, "Closest Point Search in Lattices," *IEEE Trans. Inf. Th.*, vol. 48, no. 8, pp. 2201–2214, Aug. 2002.

[23] J.D. Gibson and K. Sayood, "Lattice Quantization," *Adv. Electron. Phys.*, vol. 72, pp. 259–331, 1988.

[24] E. Viterbo and E. Biglieri, "Computing the Voronoi cell of a lattice: the diamond-cutting algorithm," *IEEE Trans. Inform. Theory*, vol. 42, no. 1, pp. 161–171, Jan. 1996.

[25] M. Antonini, M. Barlaud, and P. Mathieu, "Image coding using lattice vector quantization of wavelet coefficients," in *Proc. ICASSP*, 1991, vol. 4, pp. 2273–2276.

[26] J.-M. Moureaux, M. Antonini, and M. Barlaud, "Counting lattice points on ellipsoids: application to image coding," *Electronics Letters*, vol. 31, no. 15, pp. 1224–1225, Jul. 1995.

[27] P. Loyer, J.-M. Moureaux, and M. Antonini, "Lattice codebook enumeration for generalized gaussian source," *IEEE Trans. Inform. Theory*, vol. 49, no. 2, pp. 521–527, Feb. 2003.

[28] F. Chen, Z. Gao, and J. Villasenor, "A Lattice Vector Quantizer For Generalized Gaussian Sources," in *Proc. ICIP*, 1995, pp. 105–108.

[29] G.D. Forney, "Coset codes. I. Introduction and geometrical classification," *IEEE Trans. Inform. Theory*, vol. 34, no. 5, pp. 1123–1151, Sep. 1988.

[30] G.D. Forney, "Coset codes. II. Binary lattices and related codes," *IEEE Trans. Inform. Theory*, vol. 34, no. 5, pp. 1152–1187, Sep. 1988.

[31] T.M. Thompson, *From Error-Correcting Codes Through Sphere Packings to Simple Groups*, Number 21 in The Carus Mathematical Monographs. The Mathematical Association of America, 1983.

[32] P.L Zador, "Asymptotic Quantization Error of Continuous Signals and the Quantization Dimension," *IEEE Trans. Inform. Theory*, vol. 28, no. 2, pp. 139–148, Mar. 1982.

[33] J.H. Conway and N.J.A. Sloane, "A Lower Bound on the Average Error of Vector Quantizers," *IEEE Trans. Inform. Theory*, vol. 31, no. 1, pp. 106–109, Jan. 1985.

[34] E. Agrell and T. Eriksson, "Optimization of lattices for quantization," *IEEE Trans. Inform. Theory*, vol. 44, no. 5, pp. 1814–1828, Sep. 1998.

[35] D.J. Newman, "The Hexagon Theorem," *IEEE Trans. Inform. Theory*, vol. 28, no. 2, pp. 137–139, Mar. 1982.

[36] Z. Gao, B. Belzer, and J. Villasenor, "A comparison of the Z, E8, and Leech Lattices for Quantization of Low-shape-parameter Generalized Gaussian Sources," *IEEE Sig. Proc. Letters*, vol. 2, no. 10, pp. 197–199, Oct. 1995.

[37] G.D. Forney, R.G. Gallager, G.R. Lang, F.M. Longstaff, and S.U. Qureshi, "Efficient Modulation for Band-Limited Channels," *IEEE J. Select. Areas Commun.*, vol. 2, no. 2, pp. 158–173, Sep. 1984.

[38] J.H. Conway and N.J.A. Sloane, "Soft Decoding Techniques for Codes and Lattices, Including the Golay Code and the Leech Lattice," *IEEE Trans. Inform. Theory*, vol. 32, pp. 41–50, 1986.

[39] A. Gersho, "On the Structure of Vector Quantizers," *IEEE Trans. Inform. Theory*, vol. 28, no. 2, pp. 157–166, Mar. 1982.

[40] A. Vardy, "The Nordstrom-Robinson code: representation over GF(4) and efficient decoding," *IEEE Trans. Inform. Theory*, vol. 40, no. 5, pp. 1686–1693, Sep. 1994.

[41] A. Vardy and Y. Be'ery, "Maximum likelihood decoding of the Leech lattice," *IEEE Trans. Inform. Theory*, vol. 39, no. 4, pp. 1435–1444, Jul. 1993.

[42] K. Sayood, "A fast quantization algorithm for lattice quantizer design," in *Proc. ICASSP*, 1988, pp. 1168–1171.

[43] A.H. Banihashemi and A.K. Khandani, "On the Complexity of Decoding Lattices Using the Korkin-Zolotarev Reduced Basis," *IEEE Trans. Inform. Theory*, vol. 44, no. 1, pp. 162–171, Jan. 1998.

[44] E. Viterbo and E. Biglieri, "A Universal Lattice Code Decoder," in *GRETSI 14ième Colloque, Juans les Pins*, Sep. 1993, pp. 611–614.

[45] E. Viterbo and J. Boutros, "A Universal Lattice Code Decoder for Fading Channels," *IEEE Trans. Inform. Theory*, vol. 45, pp. 1639–1642, July 1999.

[46] U. Fincke and M. Pohst, "Improved methods for calculating vectors of short length in a lattice, including a complexity analysis," *Math. Comput.*, vol. 44, pp. 463–471, April 1985.

[47] J.-P. Adoul, "La quantification vectorielle des signaux: approche algébrique," *Ann. Télécommun.*, vol. 41, no. 3–4, pp. 158–177, 1986.

[48] T.R. Fischer, "Geometric Source Coding and Vector Quantization," *IEEE Trans. Inform. Theory*, vol. 35, no. 1, pp. 137–145, Jan. 1989.

[49] M.J. Sabin and R.M. Gray, "Product Code Vector Quantizers for Waveform and Voice Coding," *IEEE Trans. ASSP*, vol. 32, pp. 474–488, Jun. 1984.

[50] D.J. Sakrison, "A geometric treatment of the source encoding of a Gaussian random variable," *IEEE Trans. Inform. Theory*, vol. IT-14, pp. 481–486, May 1968.

[51] P. Vogel, "Analytical Coding of Gaussian Sources," *IEEE Trans. Inform. Theory*, vol. 40, no. 5, pp. 1639–1645, Sep. 1994.

[52] C. Lamblin, *Quantification vectorielle algébrique sphérique par le réseau de Barnes-Wall – Application au codage de parole*, Thèse de doctorat, Université de Sherbrooke, Québec, Canada, Mars 1988.

[53] C. Lamblin and J.-P. Adoul, "Algorithme de quantification vectorielle sphérique à partir du réseau de Gosset d'ordre 8," *Annales des Télécommunications*, vol. 43, no. 3–4, pp. 172–186, 1988.

[54] J.-P. Adoul and M. Barth, "Nearest neighbor algorithm for spherical codes from the Leech lattice," *IEEE Trans. Inform. Theory*, vol. 34, no. 5, pp. 1188–1202, Sep. 1988.

[55] J.-M. Moureaux, P. Loyer, and M. Antonini, "Low-complexity indexing method for z^n and d_n lattice quantizers," *IEEE Trans. Commun.*, vol. 46, no. 12, pp. 1602–1609, Dec. 1998.

[56] P. Rault and C. Guillemot, "Vector Indexing and Lattice Vector Quantization with Reduced or without Look-up Table," preprint IRISA, Apr. 1998.

[57] P. Rault and C. Guillemot, "Indexing Algorithms for Z_n, A_n, D_n, and D_n^{++} Lattice Vector Quantizers," *IEEE Trans. Multimedia*, vol. 3, no. 4, pp. 395–404, Dec. 2001.

[58] J. Hamkins, *Design and Analysis of Spherical Codes*, Thèse de doctorat, University of Illinois, Urbana-Champaign, 1996.

[59] P.F. Swaszek, "A Vector Quantizer for the Laplace Source," *IEEE Trans. Inform. Theory*, vol. 37, no. 5, pp. 1355–1365, Sep. 1991.

[60] J. Serra-Sagrista and J. Borrell, "Lattice points enumeration for image coding," in *IEEE Int. Conf. on Information Intelligence and Systems*, 1999, pp. 482–489.

[61] M.C. Rost and K. Sayood, "The Root Lattices as Low Bit Rate Vector Quantizers," *IEEE Trans. Inform. Theory*, vol. 34, no. 5, pp. 1053–1058, Sep. 1988.

[62] D.G. Jeong and J.D. Gibson, "Lattice vector quantization for image coding," in *Proc. ICASSP*, 1989, pp. 1743–1746.

[63] D.G. Jeong and J.D. Gibson, "Uniform and Piecewise Uniform Lattice Vector Quantization for Memoryless Gaussian and Laplacian Sources," *IEEE Trans. Inform. Theory*, vol. 39, no. 3, pp. 786–804, May 1993.

[64] T.R. Fischer and J. Pan, "Enumeration Encoding and Decoding Algorithms for Pyramid Cubic Lattice and Trellis Codes," *IEEE Trans. Inform. Theory*, vol. 41, no. 6, pp. 2056–2061, Nov. 1995.

[65] M. Barlaud, P. Sole, J.-M. Moureaux, M. Antonini, and P. Gauthier, "Elliptical codebook for lattice vector quantization," in *Proc. ICASSP*, Apr. 1993, vol. 5, pp. 590–593.

[66] M. Xie and J.-P. Adoul, "Embedded algebraic vector quantizers (EAVQ) with application to wideband speech coding," in *Proc. ICASSP*, 1996, vol. 1, pp. 240–243.

[67] S. Ragot, J.-P. Adoul, R. Lefebvre, and R. Salami, "Low complexity LSF quantization for wideband speech coding," in *Proc. IEEE Workshop on Speech Coding*, 1999, pp. 22–24.

[68] G.D. Forney, "Multidimensional constellations. II. Voronoi constellations," *IEEE J. Select. Areas Commun.*, vol. 7, no. 6, pp. 941–958, Aug. 1989.

[69] C. Pépin, *Quantification vectorielle et codage conjoint source-canal par les réseaux de points*, Thèse de doctorat, ENST, Paris, France, Déc. 1997.

[70] T.C. Chen, "A fast algorithm for uniform vector quantization," in *Proc. ICASSP*, 1987, pp. 1344–1347.

[71] C. Wang, H.Q. Cao, W. Li, and K.Z. Tzeng, "Lattice Labeling Algorithms for Vector Quantization," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 8, no. 2, pp. 206–220, Apr. 1998.

[72] K. Sayood, *Introduction to Data Compression, 2nd ed.*, Morgan Kaufman, 2000.

[73] T. Senoo and B. Girod, "Vector quantization for entropy coding of image subbands," *IEEE Trans. Image Proc.*, vol. 1, no. 4, pp. 526–533, Oct. 1992.

[74] A. Wolf and G. Rogers, "Lattice vector quantization of image wavelet coefficient vectors using a simplified form of entropy coding," in *Proc. ICASSP*, 1994, vol. V, pp. 269–272.

[75] A. Vasilache and I. Tabus, "Indexing and entropy coding of lattice codevectors," in *Proc. ICASSP*, 2001, vol. 4, pp. 2605–2608.

[76] T.R. Fischer, "Entropy-constrained geometric vector quantization for transform image coding," in *Proc. ICASSP*, 1991, pp. 2269–2272.

[77] J.-R. Ohm, "Advanced Packet Video Coding Based on Layered VQ and SBC Techniques," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 3, no. 3, pp. 208–221, June 1993.

[78] M. Xie, *Quantification vectorielle algébrique et codage de parole en bande élargie*, Thèse de doctorat, Université de Sherbrooke, Québec, Canada, Fév. 1996.

[79] K. Popat and R.W. Picard, "Cluster-based probability model applied to image restoration and compression," in *Proc. ICASSP*, 1994, vol. V, pp. 381–384.

[80] P.A. Chou, T. Lookabaugh, and R.M. Gray, "Entropy-constrained vector quantization," *IEEE Trans. on ASSP*, vol. 37, no. 1, pp. 31–42, 1989.

[81] Z. Mohd-Yusof and T.R. Fischer, "An entropy-coded lattice vector quantizer for transform and subband image coding," *IEEE Trans. on Image Proc.*, vol. 5, no. 2, pp. 289–298, Feb. 1996.

[82] M. Antonini, P. Raffy, and M. Barlaud, "Towards entropy-constrained lattice vector quantization," in *Proc. Image Processing*, 1995, vol. I, pp. 121–124.

[83] W.H. Kim, Y.H. Hu, and T.Q. Nguyen, "Wavelet-Based Image Coder with Entropy-Constrained Lattice Vector Quantization (ECLVQ)," *IEEE Trans. on Circuits & Systems II*, vol. 45, no. 8, pp. 1015–1030, Aug. 1998.

[84] P. Raffy, M. Antonini, and M. Barlaud, "Distortion-Rate Models for Entropy-Constrained Lattice Vector Quantization," *IEEE Trans. on Image Proc.*, vol. 9, no. 12, pp. 2006–2017, Dec. 2000.

[85] P. Onno and C. Guillemot, "Data-rate constrained lattice vector quantization: a new quantizing algorithm in a rate-distortion sense," in *Proc. Image Processing*, 1995, vol. 1, pp. 117–120.

[86] W.-H. Kim, "Adaptive lossless coding scheme of lattice vector quantisation," *IEE Proc. Vision, Image and Signal Processing*, vol. 146, no. 6, pp. 317–325, Dec. 1995.

[87] S.F. Simon and W. Niehsen, "On entropy coded and entropy constrained lattice vector quantization," in *Proc. Int. Conf. on Image Processing*, 1996, vol. 3, pp. 419–422.

[88] P.W. Moo and D.L. Neuhoff, "Optimal compressor functions for multidimensional companding," in *IEEE Int. Symp. on Inform. Theory*, 1997, p. 515.

[89] M.V. Eyuboglu and G.D. Forney, "Lattice and Trellis Quantization with Lattice- and Trellis-Bounded Codebooks – High-Rate Theory for Memoryless Sources," *IEEE Trans. Inform. Theory*, vol. 39, no. 1, pp. 46–59, Jan. 1993.

[90] E. Agrell, *Voronoi-Based Coding*, Thèse de doctorat, Chalmers University of Technology, Göteborg, Sweden, March 1997.

[91] N.S. Jayant and P. Noll, *Digital Coding of Waveforms – Principles and Applications to Speech and Video*, Prentice-Hall, 1984.

[92] J. Ziv, "On universal quantization," *IEEE Trans. Inform. Theory*, vol. 31, pp. 344–347, May 1985.

[93] T. Berger, *Rate Distortion Theory*, Prentice-Hall, 1971.

[94] T.D. Lookabaugh and R.M. Gray, "High-Resolution Quantization Theory and the Vector Quantizer Advantage," *IEEE Trans. Inform. Theory*, vol. 35, no. 5, pp. 1020–1033, Sep. 1989.

[95] J. Max, "Quantizing for minimum distorsion," *IRE Trans. Inform. Thero*, vol. 6, pp. 7–12, Mar. 1960.

[96] S.P. Lloyd, "Least Squares Quantizers in PCM," *IEEE Trans. Inform. Theory*, vol. 28, no. 2, pp. 129–137, Mar. 1982.

[97] P. Noll and R. Zelinski, "Bounds on quantizer performance in the low-bit rate region," *IEEE Trans. Commun.*, vol. 26, pp. 300–304, Feb. 1978.

[98] M.W. Marcellin and T.R. Fischer, "Trellis codes quantization of memoryless Gauss-Markov Sources," *IEEE Trans. Commun.*, vol. 38, pp. 82–93, Jan. 1990.

[99] H.S. Wang and N. Moayeri, "Trellis coded vector quantization," *IEEE Trans. Commun.*, vol. 40, pp. 1273–1276, Aug. 1992.

[100] R. Loria and N. Farvardin, "A structured fixed-rate vector quantizer derived from a variable-length scalar quantizer: Part I – Memoryless sources," *IEEE Trans. Inform. Theory*, vol. 39, pp. 851–867, May 1993.

[101] N. Farvardin and J.W. Modestino, "Optimum Quantizer Performance for a Class of Non-Gaussian Memoryless Sources," *IEEE Trans. Inform. Theory*, vol. 30, no. 3, pp. 485–497, May 1984.

[102] J. Pan, "Two-stage vector quantization-pyramidal lattice vector quantization and application to speech lsp coding," in *Proc. ICASSP*, 1996, vol. 2, pp. 737–740.

[103] T.R. Fischer and M. Wang, "Entropy-constrained trellis-coded quantization," *IEEE Trans. Inform. Theory*, vol. 38, pp. 415–426, Mar. 1992.

[104] T.J. Goblick and J.L. Holsinger, "Analog source digitization: A comparison of theory and practice," *IEEE Trans. Inform. Theory*, vol. 13, pp. 323–326, Apr. 1967.

[105] J. Hamkins and K. Zeger, "Gaussian Source Coding With Spherical Codes," *IEEE Trans. Inform. Theory*, vol. 48, no. 11, pp. 2980–2989, Nov. 2002.

[106] R.E. Blahut, "Computation of channel capacity and rate-distortion functions," *IEEE Trans. Inform. Theory*, vol. 18, pp. 460–473, Jul. 1972.

[107] A. Mehes and K. Zeger, "Binary lattice vector quantization with linear block codes and affine index assignments," *IEEE Trans. Inform. Theory*, vol. 44, no. 1, pp. 79–94, Jan. 1998.

[108] A.C. Hung, E.K. Tsern, and T.H. Meng, "Error-Resilient Pyramid Vector Quantization for Image Compression," *IEEE Trans. Image Proc.*, vol. 7, no. 10, pp. 1373–1386, Oct. 1998.

[109] A.C. Hung and T.H. Meng, "Multidimensional rotations for robust quantization of image data," *IEEE Trans. Imag. Proc.*, vol. 7, no. 1, pp. 1–12, Jan. 1998.

[110] V.A. Vaishampayan N.J.A. Sloane and S.D. Servetto, "Multiple-description vector quantization with lattice codebooks: design and analysis," *IEEE Trans. Inform. Theory*, vol. 47, no. 5, pp. 1718–1734, Jul. 2001.

[111] V.K. Goyal, J.A. Kelner, and J. Kovacevic, "Multiple description vector quantization with a coarse lattice," *IEEE Trans. Inform. Theory*, vol. 28, no. 3, pp. 781–788, Mar. 2002.

[112] D. Mukherjee and S.K. Mitra, "Successive refinement lattice vector quantization," *IEEE Trans. on Image Proc.*, vol. 11, no. 12, pp. 1337–1348, Dec. 2002.

[113] M. Xie and J.-P. Adoul, "Fast and low-complexity LSF quantization using algebraic vector quantizer," in *Proc. ICASSP*, 1995, vol. 1, pp. 716–720.

[114] M. Xie and J.-P. Adoul, "Algebraic vector quantization of LSF parameters with low storage and computational complexity," *IEEE Trans. on Speech and Audio Proc.*, vol. 4, no. 3, pp. 234–239, May 1996.

[115] J. Pan and T.R. Fischer, "Vector quantization-lattice vector quantization of speech LPC coefficients," in *Proc. ICASSP*, 1994, vol. 1, pp. 513–516.

[116] S. Ragot, R. Lefebvre, R. Salami, and J.-P. Adoul, "Stochastic-algebraic wideband LSF quantization," in *Proc. ICASSP*, 2000, vol. II, pp. 1169–1172.

[117] A. Vasilache, M. Vasilache, and I. Tabus, "Predictive multiple-scale lattice VQ for LSF quantization," in *Proc. ICASSP*, 1999, vol. 2, pp. 657–660.

[118] J. Pan and T.R. Fischer, "Two-Stage Vector Quantization-Lattice Vector Quantization," *IEEE Trans. Inform. Theory*, vol. 41, no. 1, pp. 155–163, Jan. 1995.

[119] J. Pan, "Two-stage Vector Quantization-Pyramidal Lattice Vector Quantization and Application to Speech LSP Coding," in *Proc. ICASSP*, 1995, vol. 2, pp. 737–740.

[120] F. Nordén, T.Z. Shabestary, and P. Hedelin, "Rate adjustable speech coding by lattice quantization," in *Proc. ICASSP*, 2003, vol. II, pp. 161–164.

[121] P. Monta and C. Shiufun, "Low rate audio coder with hierarchical filterbanks and lattice vector quantization," in *Proc. ICASSP*, 2000, vol. II, pp. 1169–1172.

[122] K. Hay, L. Mainard, and S. Saoudi, "The D5 lattice quantization for 64 kbit/s low-delay subband audio coder with a 15 kHz bandwidth," in *Proc. ICASSP*, 1997, vol. 1, pp. 319–322.

[123] H.-C. Tseng and T.R. Fischer, "Transform and Hybrid Transform/DPCM Coding of Images Using Pyramid Vector Quantization," *IEEE Trans. Commun.*, vol. 35, no. 1, pp. 79–86, Jan 1987.

[124] M. Barlaud, P. Sole, M. Antonini, and P. Mathieu, "A pyramidal scheme for lattice vector quantization of wavelet transform coefficients applied to image coding," in *Proc. ICASSP*, Mar. 1992, vol. 4, pp. 401–404.

[125] M. Khataie and M.R. Soleymani, "Indexing the Output Points of an LBVQ Used for Image Transform Coding," *IEEE Trans. Image Proc.*, vol. 9, no. 5, pp. 827–833, May 2000.

[126] H. Man, F. Kossentini, and M.J.T. Smith, "A Family of Efficient and Channel Errror Resilient Wavelet/Subband Image Coders," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 9, no. 1, pp. 95–108, Feb. 1999.

[127] P. Rault and F. Kossentini, "Robust subband image coding for wireless transmission," in *Proc. ICIP*, 1999, vol. 3, pp. 374–378.

[128] D.G. Sampson and M. Ghanbari, "Interframe coding of images using lattice vector quantization," in *Proc. Image Processing and its Applications*, 1992, pp. 131–134.

[129] E.A.B. da Silva, D.G. Sampson, and M. Ghanbari, "Image coding using successive approximation wavelet vector quantization," in *Proc. ICASSP*, 1995, vol. 4, pp. 2201–2204.

[130] P. Onno and C. Guillemot, "Wavelet packet coding with jointly optimized lattice vector quantization and data rate allocation," in *Proc. ICIP*, 1994, vol. 3, pp. 329–333.

[131] T.-C. Chen, "A lattice vector quantization using a geometric decomposition," *IEEE Trans. Commun.*, vol. 38, no. 5, pp. 704–714, May 1990.

[132] J.-M. Moureaux, P. Gauthier, M. Barlaud, and P. Bellemain, "Vector quantization of raw SAR data," in *Proc. ICASSP*, 1994, vol. 5, pp. 189–192.

[133] J. Vaisey, M. Barlaud, and M. Antonini, "Multispectral Image Coding Using Lattice VQ and the Wavelet Transform," in *Proc. ICIP*, 1998.

[134] D.G. Sampson, E.A.B. da Silva, and M. Ghanbari, "Wavelet lattice quantization for low bit rate video coding," in *Proc. ICC*, 1995, vol. 3, pp. 1423–1427.

[135] L. Lois and S. Bozoki, "Transcoding of MPEG video using lattice vector quantization," in *Proc. ICIP*, 1998, vol. 2, pp. 341–345.

[136] H. Man, R.L de Queiroz, and M.J.T. Smith, "Three-dimensional subband coding techniques for wireless video communications," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 12, no. 6, pp. 386–397, 2002.

[137] R.A. Silverman and M. Balser, "Coding for a constant data rate source," *IRE Trans. Inform. Theory*, vol. 4, pp. 50–63, 1954.

[138] T. Berger, F. Jelinek, and J.K. Wolf, "Permutation codes for sources," *IEEE Trans. Inform. Theory*, vol. 18, pp. 160–169, 1972.

[139] J.P.N. Schalkwijk, "An algorithm for source coding," *IEEE Trans. Inform. Theory*, vol. 18, no. 3, pp. 395–399, May 1972.

[140] T.R. Fischer and R.M. Dicharry, "Vector Quantizer Design for Memoryless Gaussian, Gamma, and Laplacian Source," *IEEE Trans. Commun.*, vol. 32, no. 9, pp. 1065–1069, Sep. 1984.

Chapitre 3

CODAGE DE VORONOÏ QUASI-ELLIPSOÏDAL

Ce chapitre traite du problème de la conception de quantificateurs vectoriels algébriques ellipsoïdaux. On cherche en particulier à définir des dictionnaires C de forme ellipsoïdale, conduisant à un bon compromis entre performances de quantification et complexité. Une façon de définir C consiste à tronquer un réseau de points Λ de $\mathbb{R}^N$ par un ellipsoïde. Un exemple est donné à la figure 3.1 dans le cas du réseau A_2.

Les dictionnaires C sont définis ici de la même façon en ne retenant qu'un sous-ensemble fini d'un réseau de points Λ. Pour les mettre en œuvre, on doit trouver des algorithmes efficaces d'indexation et de recherche. L'indexation implique de pouvoir numéroter tous les éléments de C par des indices binaires distincts et de retrouver chaque élément de C à partir de son indice, tandis que la recherche dans C implique idéalement de trouver le plus proche voisin de n'importe quel point $\mathbf{x}$ de $\mathbb{R}^N$ dans C. On définit dans ce chapitre une technique générale et paramétrable de quantification algébrique (quasi)-ellipsoïdale, qui conduit à une indexation totalement algorithmique, rapide et indépendante de la taille de C, et virtuellement sans stockage, ainsi qu'à une recherche dont la complexité maximale est limitée. La quantification est dite quasi-ellipsoïdale, car la région de troncature utilisée n'est pas

un ellipsoïde, bien qu'elle s'en approche.

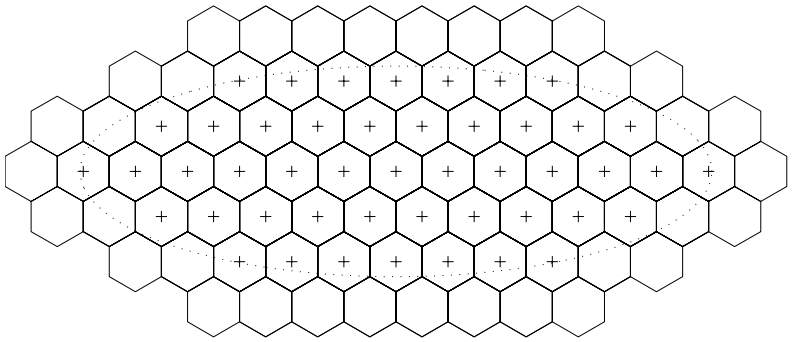


Figure 3.1: Exemple de dictionnaire de quantification ellipsoïdal C dans le cas du réseau A_2.

La technique de quantification développée dans ce chapitre généralise le codage de Voronoï, introduit par Conway et Sloane dans [1]. Elle est donc appelée codage de Voronoï quasi-ellispoïdal. Elle a été partiellement introduite dans [2, chap. 1]. On apporte néanmoins des éléments nouveaux et importants qui complètent l'étude de [2]. Tout d'abord, les dictionnaires de quantification C utilisés dans [2] sont clairement définis ici comme des codes de Voronoï, alors qu'ils sont introduits dans [2] uniquement par des algorithmes d'indexation. Ils sont définis ici de la façon la plus générale possible, contrairement à [2]. De plus, on introduit un algorithme d'indexation général (indépendant du réseau Λ utilisé), qui remplace les algorithmes spécifiques de [2]. Enfin on présente un algorithme de recherche du plus proche voisin basé sur une recherche rapide de vecteur pertinent. Cet algorithme, gérant la surcharge par projection et réduction itérative, a une complexité maximale limitée et indépendante de la source – contrairement à l'algorithme de recherche décrit dans [2]. L'utilisation d'une projection avant réduction itérative est inspirée par les développements de [3].

Les développements présentés ici ont été motivés par une approche semi-paramétrique de quantification d'enveloppe de prédiction linéaire [4]. Suivant cette approche, les coefficients de prédiction linéaire (ou une représentation équivalente) sont codés à l'aide d'une analyse en composantes principales [2, 4] ou en modélisant la distribution de probabilité de la source par mélange de lois gaussiennes [4, 5, 6, 7, 8, 9] ; le problème de la quantification est alors essentiellement réduit à la conception des codes pour représenter des sources gaussiennes vectorielles.

Des codes algébriques ellipsoidaux ont été proposés dans [10, 11, 12]. Néanmoins, ces codes possèdent certaines limites. Ainsi, le réseau $\mathbb{Z}$ utilisé dans [10] n'apporte aucun gain granulaire. De plus, la technique de [11, 12], qui généralise la troncature sphérique en énumérant des points sur des orbites elliptiques, a une complexité exponentielle en fonction du débit du code et de la dimension de quantification, et requiert une conception et une mise en œuvre non-triviales et dépendantes du débit de codage.

Ce chapitre est organisé comme suit. Des définitions préliminaires et une revue succinte du codage de Voronoï sont présentées dans la section 3.1. Les codes de Voronoï quasi-ellipsoidaux sont définis et étudiés dans la section 3.2. Des algorithmes d'indexation généraux sont détaillés dans la section 3.3. Ceux-ci comprennent une étape qui peut être optimisée en fonction du réseau Λ utilisé ; on détaille ici cette optimisation dans le cas de réseaux A_2, D_N ($N \geq 2$), $2D_N^+$ (N pair ≥ 4) et Λ_{16}. Le problème de la recherche du plus proche voisin est étudié dans la section 3.4. L'optimisation des performances pour la source est présentée à la section 3.5. En particulier, le codage de Voronoï est optimisé dans un schéma de type gain-forme ; cependant, cette optimisation ne rentre pas dans le cadre de la théorie de la quantification vectorielle de type gain-forme de [13]. Les performances et la complexité des codes de Voronoï sont discutées dans la section 3.6, avant de conclure dans la section 3.7.

3.1 Préliminaires

On reprend et complète ici une partie des définitions de base vues au chapitre 2.

3.1.1 Réseaux de points

On considère ici uniquement des réseaux réguliers de points dans $\mathbb{R}^N$ de rang maximal. En général, un réseau dans $\mathbb{R}^N$ est noté Λ et défini par :

$$\Lambda = \left\{ k_1\mathbf{v}_1 + \cdots + k_N\mathbf{v}_N | \mathbf{k} = (k_1, \cdots, k_N) \in \mathbb{Z}^N \right\} \tag{3.1}$$

où $\{\mathbf{v}_i\}_{1\leq i\leq n}$ forme un ensemble de vecteurs linéairement indépendants dans $\mathbb{R}^N$. Ces vecteurs de base forment une matrice appelée matrice génératrice $M(\Lambda)$ de Λ :

$$M(\Lambda) = \begin{bmatrix} \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_N \end{bmatrix}. \tag{3.2}$$

Ils correspondent aux lignes de $M(\Lambda)$. Ainsi, si $\mathbf{k}$ est un vecteur dans $\mathbb{Z}^N$, $\mathbf{x} = \mathbf{k}M(\Lambda)$ génère un point dans Λ et $\mathbf{x}M(\Lambda)^{-1}$ fournit la décomposition $\mathbf{k}$ dans la base adoptée. Le réseau Λ peut donc être écrit :

$$\Lambda = \left\{\mathbf{k}M(\Lambda)|\mathbf{k} \in \mathbb{Z}^N\right\}. \tag{3.3}$$

Dans un tel réseau toutes les régions de Voronoï sont congruentes [14] et on ne retient ici que la région de Voronoï associée à l'origine, laquelle est notée $V(\Lambda)$.

On utilise plus spécifiquement pour Λ les réseaux A_2, D_N, $2D_N^+$ and Λ_{16}. Ceux-ci sont tous définis et spécifiés par une matrice génératice dans [14]. Cependant pour lever toute ambiguïté, les matrices génératrices utilisées ci-après sont toutes définies dans les annexes 2.A et 3.A. Les matrices utilisées ici sont toutes triangulaires inférieures ; cette propriété est imposée ici pour simplifier les algorithmes d'indexation des codes de Voronoi quasi-ellipsoïdaux.

Suivant [15], si Λ' est un sous-réseau de Λ, $|\Lambda/\Lambda'|$ fait référence à l'ordre de la partition Λ/Λ'. On trouve plus de détails à ce sujet au paragraphe 2.2.4.

3.1.2 Troncature par région de Voronoï : codage de Voronoï

Étant donné un réseau Λ dans $\mathbb{R}^N$, un code de Voronoï C peut être défini comme dans [1]

$$C = \Lambda \cap (m\,V(\Lambda) + \mathbf{a}), \tag{3.4}$$

où m est un entier ≥ 2 et $\mathbf{a}$ est un décalage approprié dans $\mathbb{R}^N$ qui assure qu'aucun point de Λ ne soit sur la frontière de la région $m\,V(\Lambda) + \mathbf{a}$.

L'équation 3.4 montre que le code C est obtenu en ne retenant de Λ que les points qui appartiennent à la région $m\,V(\Lambda)+\mathbf{a}$. La région de troncature de Λ est $m\,V(\Lambda)+\mathbf{a}$. Puisque que cette région a la forme de $V(\Lambda)$, la troncature utilisée peut être appelé troncature de Voronoï. Un exemple de code C est montré à la figure 3.2 (a) dans le cas du réseau $\Lambda = A_2$, pour $m = 4$ et $\mathbf{a} = (0.3, 0)$.

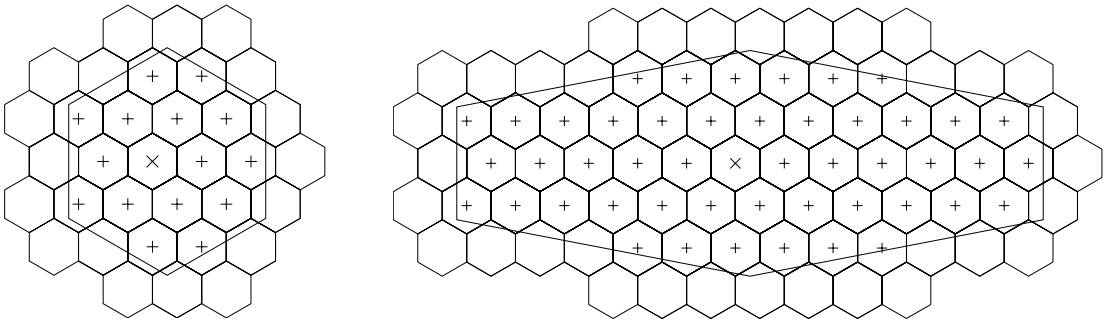
La taille du code C est $|\Lambda/m\Lambda| = m^N$ [1] et le débit de C est de $\log_2 m \geq 1$ bit par dimension. En choisissant $m = 2^R$ où R est entier ≥ 1, on obtient des débits entiers par dimension. Le code de la figure 3.2 (a) a ainsi un débit de 2 bits par dimension et contient 16 points.

La troncature de Voronoï telle qu'introduite dans [1] est quasi-sphérique. Elle a été étendue dans [16] au cas où Λ est tronqué par la région de Voronoï d'un réseau géométriquement similaire à Λ. Elle conduit à des codes algébriques optimisés pour une indexation rapide. Les algorithmes d'indexation associés reposent essentiellement sur le décodage du réseau et l'opération de modulo [1].

Un code de Voronoï peut être interprété de plusieurs façons :

- D'après l'équation 3.4, un code de Voronoï peut être vu directement comme la troncature de Λ par une région de Voronoï $V(\Lambda)$ dilatée par un facteur m et translatée de $\mathbf{a}$.
- On peut également interpréter la région de troncature comme étant $V(m\Lambda)+\mathbf{a}$ (au lieu de $mV(\Lambda)+\mathbf{a}$) où $m\Lambda$ est un réseau de troncature [16]. L'avantage de cette interprétation vient du fait que C peut alors être vu comme un ensemble de vecteurs générateurs (ou *coset leaders*) de norme minimale de la partition $\Lambda/m\Lambda$ [16].
- La troncature de Voronoï peut être interprétée de façon équivalente comme une congruence [14] – les réseaux Λ et $m\Lambda$ étant les opérandes.

Le problème de l'optimisation du décalage $\mathbf{a}$ a été abordé dans [1, 16]. Dans [1], $\mathbf{a}$ est ajusté de sorte qu'il soit le barycentre de C, et donc que le code $C-\mathbf{a}$ soit à moyenne nulle. Une procédure itérative est définie pour optimiser $\mathbf{a}$: $\mathbf{a}$ est d'abord initialisé de façon aléatoire en s'assurant qu'aucun point de Λ n'est sur la frontière de $mV(\Lambda)+\mathbf{a}$; à chaque itération, $\mathbf{a}$ est remplacé par



(a) code de Voronoï
$m = 4$, $\mathbf{a} = (0.3, 0)$

(b) code de Voronoï quasi-ellipsoïdal
$\mathbf{m} = (12, 4)$, $\mathbf{a} = (0.3, 0)$

Figure 3.2: Exemple de codes de Voronoï pour $\Lambda = A_2$.

le centroïde de C. Cette procédure minimise de façon heuristique l'énergie moyenne du code C [1]. Des méthodes d'optimisation alternatives du vecteur $\mathbf{a}$ sont décrites dans [16] :

- Une méthode fixe d'abord $\mathbf{a} = (0, \cdots, 0)$ (des points de Λ sont alors sur la frontière de $mV(\Lambda) + \mathbf{a}$). Puis elle identifie tous les vecteurs de norme minimale dans chacun des cosets de $m\Lambda$ dans Λ et sélectionne un représentant par coset suivant une relation d'ordre [16, p. 942]. Cette méthode tend à minimiser l'énergie moyenne du code $C - \mathbf{a}$.

- L'autre méthode fixe le décalage à $\mathbf{a} = (0, \cdots, 0)$. Elle implique de coder une information supplémentaire à débit variable [16, p. 943].

3.2 Codes de Voronoï quasi-ellipsoidaux

3.2.1 Définition

Étant donné un réseau Λ dans $\mathbb{R}^N$, un code de Voronoï quasi-ellipsoïdal C est défini comme :

$$C = \Lambda \cap \left(V(\Lambda') + \mathbf{a} \right), \tag{3.5}$$

où

$$\Lambda' = \left\{ (m_1 x_1, \cdots, m_N x_N) | \mathbf{x} \in \Lambda, \mathbf{m} \in (\mathbb{N} \setminus \{0,1\})^N \right\}, \tag{3.6}$$

le vecteur $\mathbf{m}$ étant de plus contraint de telle sorte que Λ' soit un sous-réseau de Λ. Si Λ est un réseau de points associé à un empilement de sphères, Λ' est un réseau de points associé à un empilement d'ellipsoïdes ; la région de Voronoï $V(\Lambda')$ n'est autre que $V(\Lambda)$ dilatée dimension par dimension suivant les composantes de $\mathbf{m}$.

Par définition le vecteur $\mathbf{m}$ possède N composantes entières ≥ 2. Le décalage $\mathbf{a}$ dans $\mathbb{R}^N$ joue le même rôle que dans le codage de Voronoï standard. La région $V(\Lambda')$, qui est la région de Voronoï de Λ' associée à l'origine, correspond à la région $V(\Lambda)$ mise à l'échelle dimension par dimension suivant les composantes de $\mathbf{m}$. Cette troncature de réseau est illustrée à la figure 3.2 (b), pour $\mathbf{m} = (12, 4)$ et $\mathbf{a} = (0.3, 0)$.

Propriété: La taille du code C est $\prod_{i=1}^{N} m_i$
Preuve: Si Λ' est un sous-réseau de Λ, la taille de C est $|\Lambda/\Lambda'|$ (voir par exemple [16]). De plus il est facile de vérifier d'après la définition de Λ' que

$$\Lambda = \bigcup_{\substack{\mathbf{k} \in \mathbb{Z}^N \\ \text{tel que } 0 \leq k_i < m_i \ \forall i=1,2,\cdots,n}} \Lambda' + \mathbf{k}M(\Lambda) \tag{3.7}$$

Par conséquent il y a $\prod_{i=1}^{N} m_i$ cosets de Λ' dans Λ, ce qui prouve la propriété.

3.2.2 Modulos de Voronoï admissibles et matrice Q

On se propose ici de trouver tous les vecteurs $\mathbf{m}$ dans $(\mathbb{N}\setminus\{0,1\})^N$ tels que le réseau Λ', défini par l'équation 3.6, soit un sous-réseau de Λ.

Théorème : Λ' est un sous-réseau de Λ si et seulement si la matrice

$$Q = M(\Lambda) diag(m_1, \cdots, m_N) M(\Lambda)^{-1} \tag{3.8}$$

est une matrice à coefficients entiers.

Preuve: On suppose d'abord que Λ' est un sous-réseau de Λ. Les lignes de $M(\Lambda)$ sont dans Λ, et les lignes de $M(\Lambda) diag(m_1, \cdots, m_N)$ sont dans Λ'. Puisque Λ' est un sous-réseau de Λ, les lignes de $M(\Lambda) diag(m_1, \cdots, m_N)$ sont aussi dans Λ. Chaque ligne de $M(\Lambda) diag(m_1, \cdots, m_N)$ multipliée à droite par $M(\Lambda)^{-1}$ donne donc une décomposition dans la base de Λ en terme de coordonnées entières. Par conséquent $M(\Lambda) diag(m_1, \cdots, m_N) M(\Lambda)^{-1}$ est une matrice à coefficients entiers.

Pour prouver la réciproque, on suppose que $M(\Lambda) diag(m_1, \cdots, m_N) M(\Lambda)^{-1}$ est une matrice à coefficients entiers. Un point λ' dans Λ' peut être décomposé comme $\lambda' = \mathbf{k} M(\Lambda')$. On peut ré-écrire cette dernière expression sous la forme $\lambda' = \mathbf{k} M(\Lambda) diag(m_1, \cdots, m_N)$ d'après la définition de Λ'. Par hypothèse, λ' étant un point de Λ', c'est aussi un point de Λ. Donc $\Lambda' M(\Lambda)^{-1}$ est un vecteur entier, et λ' est obtenu par une combinaison linéaire à coefficients de proportionalité entiers de la base de Λ. Par suite $\lambda' \in \Lambda$ et Λ' est un sous-réseau de Λ. □

Ce théorème se comprend intuitivement en remarquant que $M(\Lambda) diag(m_1, \cdots, m_N)$ donne en fait une base de Λ' et que la matrice Q fournit les coordonnées de cette base de Λ' dans la base de Λ spécifiée par $M(\Lambda)$. Les conditions sur $\mathbf{m}$ pour que Λ' soit un sous-réseau de Λ se déduisent du théorème ci-dessus. Le critère est de plus invariant pour toute transformation orthogonale entière de Λ : si $M(\Lambda)$ est une matrice génératrice de Λ, toute matrice de la forme $\alpha T M(\Lambda)$ spécifiant $\alpha\Lambda$ où α est un facteur d'échelle et T une matrice orthogonale entière ($\det T = \pm 1$), conduit à des contraintes identiques sur $\mathbf{m}$. Cependant, en général deux matrices génératices $M(\Lambda_1)$ et $M(\Lambda_2)$ de deux réseaux équivalents Λ_1 et Λ_2 ne donnent pas les mêmes contraintes sur $\mathbf{m}$. Par exemple, les contraintes sur $\mathbf{m}$ diffèrent pour les deux réseaux équivalents $\Lambda_1 = \mathbb{Z}^2$ et $\Lambda_2 = D_2$.

Corollaire: Les valeurs de $\mathbf{m}$ admissibles sont des points de $(\mathbb{N}\backslash\{0,1\})^N \cap A$, où A est un réseau régulier entier fonction de Λ.

Preuve: On note par A l'ensemble des vecteurs $\mathbf{m}$ de $\mathbf{R}^N$ tels que $Q = M(\Lambda)diag(m_1, \cdots, m_N)M(\Lambda)^{-1}$ soit une matrice entière. Sans perte de généralité, on impose à $M(\Lambda)$ d'être triangulaire inférieure. Alors il est facile de montrer que Q est triangulaire inférieure et a comme coefficients diagonaux $m_1, \cdots, m_N$, i.e.

$$Q = \begin{bmatrix} m_1 & & & \\ \alpha_{2,1} & m_2 & & \\ \vdots & \ddots & \ddots & \\ \alpha_{N,1} & \cdots & \alpha_{N,N-1} & m_N \end{bmatrix}. \tag{3.9}$$

On en déduit que si $\mathbf{m} \in A$, alors $\mathbf{m}$ est un vecteur entier. De plus, on peut vérifier que l'ensemble A est non-vide et qu'il admet une structure de groupe additif. Par conséquent, A est un réseau de points entier, c'est-à-dire un sous-réseau de $\mathbb{Z}^N$. La preuve est complétée en remarquant que les vecteurs admissibles $\mathbf{m}$ sont définis dans $(\mathbb{N}\backslash\{0,1\})^N \cap A$. □

3.2.3 Exemples de contraintes sur m

Pour le réseau A_2, la condition devient

$$Q = M(A_2) \begin{bmatrix} m_1 & 0 \\ 0 & m_2 \end{bmatrix} M(A_2)^{-1} = \begin{bmatrix} m_1 & 0 \\ \frac{m_1 - m_2}{2} & m_2 \end{bmatrix}, \tag{3.10}$$

où $M(A_2)$ est donnée à l'annexe 3.A. On en déduit que m_1 et m_2 doivent être de parité identique (paire ou impaire). La même approche peut être utilisée pour les réseaux D_N, $2D_N^+$ et Λ_{16}, en utilisant les matrices génératrices définies à l'annexe 3.A. Les résultats sont rassemblés dans le tableau 3.1.

3.3 Algorithmes d'indexation

En général l'indexation d'un code issu d'un réseau de points de Λ est non-triviale. Le codage de Voronoï, tel qu'introduit dans [1], est une technique élégante de troncature de réseau conduisant à des algorithmes d'indexation généraux et simples à mettre en œuvre. Il est souhaitable de retenir cet

TABLEAU 3.1: Modulos de Voronoï $\mathbf{m}$ admissibles (par définition $\mathbf{m}$ est aussi contraint à être dans $(\mathbb{N}\backslash\{0,1\})^N$).

Réseau	Constraintes sur $\mathbf{m}$
A_2	m_1 et m_2 ont une parité identique (i.e. $\mathbf{m} \in D_2$)
D_N	$m_1, \cdots, m_N$ sont de même parité (i.e. $\mathbf{m} \in D_N^*$)
$2D_N^+$	$m_1, \cdots, m_N$ sont de même parité et $\sum_{i=1}^N m_i$ est un multiple de 4 (si n est pair, $\mathbf{m} \in 2D_N^+$)
Λ_{16}	$\mathbf{m} \in \Lambda_{16}$

attribut pour les codes de Voronoï quasi-ellipsoïdaux, et de minimiser les changements à l'algorithme original de [1]. On montre ici que, pourvu que $\mathbf{m}$ soit un modulo de Voronoï admissible, un code de Voronoï quasi-ellipsoïdal défini dans un réseau Λ et spécifié par $\mathbf{m}$ et $\mathbf{a}$ peut être indexé à partir des algorithmes détaillés dans les paragraphes suivants et résumés à la figure 3.3. Les opérations vectorielles **mod**, **mult** and **div** sont respectivement la multiplication, la division et le modulo composante par composante. Pour $\mathbf{x}$ et $\mathbf{y}$ dans $\mathbb{R}^N$, $\mathbf{j}$ dans $\mathbb{Z}^N$ et $\mathbf{m}$ dans $(\mathbb{N}\backslash\{0,1\})^N$, on a :

$$\mathbf{mod}(\mathbf{j}, \mathbf{m}) = (j_1 (\mathrm{mod}\, m_1), \cdots, j_N (\mathrm{mod}\, m_N)) \tag{3.11}$$

$$\mathbf{mult}(\mathbf{x}, \mathbf{y}) = (x_1 y_1, \cdots, x_N y_N) \tag{3.12}$$

$$\mathbf{div}(\mathbf{x}, \mathbf{y}) = (x_1/y_1, \cdots, x_N/y_N) \tag{3.13}$$

où mod est le modulo scalaire.

L'indice $\mathbf{k}$ n'est pas scalaire, mais plutôt un vecteur à composantes entières vérifiant $0 \leq k_i < m_i$ pour $i \in \{1, \cdots, N\}$. Par conséquent, $\mathbf{k}$ peut prendre $\prod_{i=1}^N m_i$ valeurs différentes. Pour former un indice scalaire K, les composantes d'un indice de Voronoï $\mathbf{k}$ peuvent être multiplexées (exemple : $K = k_1 \prod_{i=2}^N m_i + k_2 \prod_{i=3}^N m_i + \cdots + k_{N-1} m_N + k_N$). L'indice K peut être représenté sur $\lceil \log_2 \left(\prod_{i=1}^N m_i \right) \rceil$ bits.

Afin de mieux expliquer le principe de ces algorithmes, le décodage est décrit avant le codage.

Codage: mot de code $\lambda \in \Lambda \rightarrow$ indice $\mathbf{k}$

1. Calculer $\mathbf{j} = \lambda M(\Lambda)^{-1}$
2. Modifier conditionellement $\mathbf{j}$: $\mathbf{j}' = \mathbf{j} + f(\mathbf{j}, \mathbf{m})$
3. Calculer $\mathbf{k} = \mathbf{mod}(\mathbf{j}', \mathbf{m})$

Décodage: indice $\mathbf{k} \rightarrow$ mot de code $\lambda \in \Lambda$

1. Calculer $\mathbf{x} = \mathbf{k}M(\Lambda)$
2. Calculer $\mathbf{y} = \mathbf{div}(\mathbf{x} - \mathbf{a}, \mathbf{m})$
3. Trouver le plus proche voisin $\mathbf{z}$ de $\mathbf{y}$ dans Λ
4. Calculer $\lambda = \mathbf{x} - \mathbf{mult}(\mathbf{m}, \mathbf{z})$

Figure 3.3: Algorithmes d'indexation pour un code de Voronoï quasi-ellipsoïdal.

3.3.1 Décodage

L'algorithme de décodage de la figure 3.3 associe à un indice $\mathbf{k}$ un mot de code λ. Cet algorithme est une simple généralisation de [1]. On peut vérifier[1] que pour que le décodage produise des points dans Λ, Λ' doit être sous-réseau de Λ. Les étapes 1 à 4 de l'algorithme sont résumées ci-dessous :

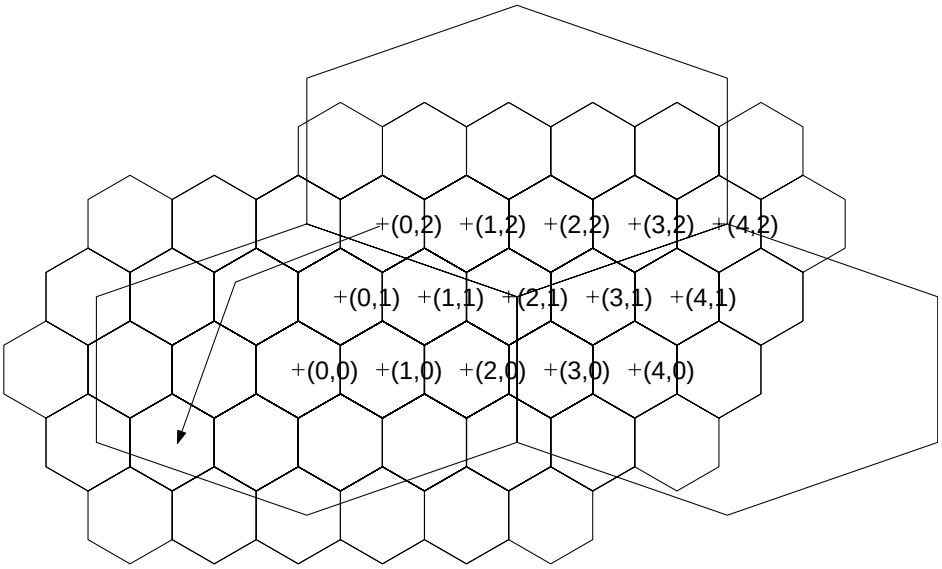
$$\lambda = \mathbf{x} - Q_{\Lambda'}(\mathbf{x} - \mathbf{a}) \tag{3.14}$$

où $Q_{\Lambda'}$ désigne le décodage dans Λ' et $\mathbf{x} = \mathbf{k}M(\Lambda)$. D'après l'équation 3.14, tout λ généré par l'algorithme de décodage est dans C.

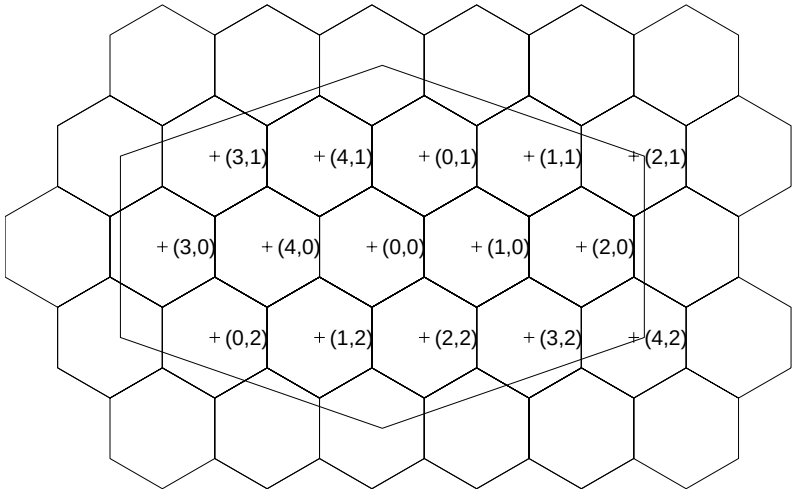
Le fonctionnement de l'algorithme de décodage est illustré à la figure 3.4. Dans cet exemple, le réseau est A_2 et les paramètres du codage de Voronoï quasi-ellipsoïdal sont $\mathbf{a} = (0.1, 0)$ et $\mathbf{m} = (5, 3)$. La figure 3.4 (a) rassemble tous les points $\mathbf{x}$ générés à l'étape 1 ; ces points sont étiquetés par les indices $\mathbf{k} = (k_1, k_2)$ correspondants. Les étapes 2 à 4 consistent à identifier dans quelle "grande cellule" (c'est-à-dire quelle région de Voronoï de $\Lambda' + \mathbf{a}$) se situe un $\mathbf{x}$ donné et de ramener $\mathbf{x}$ à l'intérieur de la région de troncature. Cette opération est illustrée par une flèche. L'ensemble des mots de code décodés de la sorte est représenté à la figure 3.4 (b). L'origine correspond au point étiqueté par $(0, 0)$.

À partir de cet exemple, il est facile de vérifier intuitivement que deux indices différents $\mathbf{k}$ et

[1] Voir l'étape 4 du décodage : le terme $\mathbf{mult}(\mathbf{m}, \mathbf{z})$ doit être dans Λ pour que λ soit aussi dans Λ.



(a) ensemble des $\mathbf{x} = \mathbf{k}M(\Lambda)$



(b) ensemble des mots de code λ étiquetés par un indice $\mathbf{k}$

Figure 3.4: Exemple d'indexation d'un code de Voronoï quasi-ellipsoïdal (cas de A_2).

$\mathbf{k}'$ génèrent par décodage deux mots de code différents dans C. Cette propriété est la conséquence directe de l'équation 3.14, laquelle peut être interprétée comme une opération de modulo sur le réseau Λ. Deux points de Λ se trouvant dans un même région de Voronoï de $\Lambda' + \mathbf{a}$ correspondent

sans équivoque à deux points congruents dans une autre région de Voronoï de $\Lambda' + \mathbf{a}$.

3.3.2 Codage

L'algorithme de codage de la figure 3.3 associe un indice $\mathbf{k}$ à un mot de code λ de C. Si les étapes 1 et 3 généralisent le codage de [1], l'étape 2, appelée "modification conditionnelle", constitue une spécificité du codage de Voronoï quasi-ellipsoïdal par rapport à [1].

Principe de l'algorithme de codage : Un élément λ d'un code de Voronoï quasi-ellipsoïdal C specifié dans Λ par $\mathbf{m}$ et $\mathbf{a}$ peut être interprété comme un mot de code généré en décodant un indice $\mathbf{k}$. L'objectif est de retrouver ce même indice $\mathbf{k}$ à partir de λ.

En utilisant les étapes 1 et 4 de l'algorithme de décodage, on peut postuler qu'il existe un indice $\mathbf{k}$ et un point $\mathbf{z} \in \Lambda$ tels que :

$$\lambda = \mathbf{k}M(\Lambda) - \mathbf{mult}(\mathbf{m}, \mathbf{z}). \tag{3.15}$$

Puisque $\mathbf{z} \in \Lambda$, il existe aussi $\mathbf{l} \in \mathbb{Z}^N$ tel que $\mathbf{z} = \mathbf{l}M(\Lambda)$. Alors l'équation 3.15 devient :

$$\lambda = \mathbf{k}M(\Lambda) - \mathbf{l}M(\Lambda)diag(m_1, \cdots, m_N). \tag{3.16}$$

Par conséquent, l'indice intermédiaire $\mathbf{j} = \lambda M(\Lambda)^{-1}$ à l'étape 1 de l'algorithme de codage peut être décomposé sous la forme :

$$\mathbf{j} = \mathbf{k} - \mathbf{l}Q, \tag{3.17}$$

où

$$Q = M(\Lambda)diag(m_1, \cdots, m_N)M(\Lambda)^{-1}. \tag{3.18}$$

Pour obtenir automatiquement l'indice $\mathbf{k}$ à partir de λ, le terme additionnel $\mathbf{l}Q$ doit être "supprimé". C'est le rôle des étapes 2 et 3 de l'algorithme de codage.

Principe de la modification conditionnelle : Si $M(\Lambda)$ est spécifiée comme une matrice triangulaire inférieure, il est facile de vérifier que la matrice Q est aussi triangulaire inférieure et a pour

éléments diagonaux $m_1, \cdots, m_N$, comme décrit ci-dessous :

$$Q = \begin{bmatrix} m_1 & & & \\ \alpha_{2,1} & m_2 & & \\ \vdots & \ddots & \ddots & \\ \alpha_{N,1} & \dots & \alpha_{N,N-1} & m_N \end{bmatrix} \tag{3.19}$$

Cette matrice a été vue au paragraphe 3.2.2, à l'équation 3.9. De plus, si $\mathbf{m}$ est contraint de telle sorte que Λ' soit un sous-réseau de Λ, Q est entière. En utilisant l'équation 3.19, on peut reformuler l'équation 3.17 comme un système d'équations :

$$\begin{cases} j_1 = k_1 - (m_1 l_1 + \alpha_{2,1} l_2 + \cdots + \alpha_{N,1} l_N) \\ j_2 = k_2 - (m_2 l_2 + \alpha_{3,2} l_3 + \cdots + \alpha_{N,2} l_N) \\ \vdots \\ j_{N-1} = k_{n-1} - (m_{N-1} l_{N-1} + \alpha_{N,N-1} l_N) \\ j_N = k_N - m_N l_N \end{cases} \tag{3.20}$$

En utilisant le fait que la $i^{ème}$ composante de $\mathbf{k}$ est bornée, soit $0 \leq k_i < m_i$, on peut calculer récursivement :

$$\begin{cases} l_N = -\lfloor j_N / m_N \rfloor \\ l_{N-1} = -\lfloor (j_{n-1} + \alpha_{N,N-1} l_N)/m_{N-1} \rfloor \\ \vdots \\ l_2 = -\lfloor (j_2 + (\alpha_{3,2} l_3 + \cdots + \alpha_{N,2} l_N)/m_2 \rfloor \end{cases} \tag{3.21}$$

où $\lfloor . \rfloor$ désigne l'arrondi à l'entier inférieur. Ensuite en calculant

$$\begin{cases} j'_N = j_N \\ j'_{N-1} = j_{N-1} + \alpha_{N,N-1} l_N \\ \vdots \\ j'_2 = j_2 + (\alpha_{3,2} l_3 + \cdots + \alpha_{N,2} l_N) \\ j'_1 = j_1 + (\alpha_{2,1} l_2 + \cdots + \alpha_{N,1} l_N) \end{cases} \tag{3.22}$$

on peut vérifier que $j'_i = k_i \pmod{m_i}$ pour tout $i \in \{1, \cdots, n\}$. La modification conditionnelle est donnée par les équations 3.21 et 3.22.

On rappelle que pour que la modification conditionnelle soit applicable, le vecteur **m** doit être un modulo de Voronoï admissible ; dans ce cas, la matrice Q, definie à l'équation 3.18, est entière.

3.3.3 Exemples de modification conditionnelle

On présente ici des réalisations spécifiques de l'algorithme de modification conditionnelle pour les réseaux A_2, D_N, $2D_N^+$ et Λ_{16}.

Modification conditionnelle pour A_2 : Dans le cas de A_2 la modification conditionnelle devient :

---

i. Calculer $l_2 = -\lfloor j_2/m_2 \rfloor$

ii. Fixer $j_1' = j_1 + (m_1 - m_2)l_2/2$ et $j_2' = j_2$

Pour prouver que cette procédure est valide, on peut d'abord observer que la matrice Q de l'équation 3.18 devient pour A_2 :

$$Q = \begin{bmatrix} m_1 & 0 \\ \frac{m_1 - m_2}{2} & m_2 \end{bmatrix} \tag{3.23}$$

On retrouve ici l'équation 3.10. Le système donné à l'équation 3.20 se réduit à :

$$\begin{cases} j_1 = k_1 - m_1 l_1 - (m_1 - m_2)l_2/2 \\ j_2 = k_2 - m_2 l_2 \end{cases} \tag{3.24}$$

Par suite, $j_2 \pmod{m_2} = k_2$. En général on a $j_1 \pmod{m_1} \neq k_1$ à cause du terme $(m_1 - m_2)l_2/2$. Si **m** est un modulo de Voronoï admissible, Q est entière, donc $(m_1 - m_2)/2$ est entier, donc $(m_1 - m_2)l_2/2$ aussi.

En utilisant le fait que par définition $0 \leq k_2 < m_2$, on peut calculer l'entier l_2 à partir j_2 tel que $l_2 = -\lfloor j_2/m_2 \rfloor$. Alors en posant $j_1' = j_1 + (m_1 - m_2)l_2/2$ et $j_2' = j_2$, il vient $j_1' = k_1 \pmod{m_1}$ et $j_2' = k_2 \pmod{m_2}$.

Modification conditionnelle pour D_N : On peut utiliser le même principe que dans le cas de A_2. Cependant cette fois-ci la matrice Q est donnée par :

$$Q = \begin{bmatrix} m_1 & & & \\ \frac{m_1-m_2}{2} & m_2 & & \\ \vdots & & \ddots & \\ \frac{m_1-m_N}{2} & & & m_N \end{bmatrix}, \tag{3.25}$$

où les éléments manquants dans Q sont des zéros. On obtient donc la suite d'opérations pour D_N :

i.	Calculer $l_i = -\lfloor j_i/m_i \rfloor$ pour $i \in \{2, \cdots, N\}$
ii.	Fixer $j_1' = j_1 + \sum_{i=2}^{N} (m_1 - m_i) l_i/2$
	Fixer $j_i' = j_i$ for $i \in \{2, \cdots, n\}$

Modification conditionnelle pour $2D_N^+$: Pour $2D_N^+$ la matrice Q est donnée ci-dessous par :

$$Q = \begin{bmatrix} m_1 & & & & \\ \frac{m_1-m_2}{2} & m_2 & & & \\ \vdots & & \ddots & & \\ \frac{m_1-m_{N-1}}{2} & & & m_{N-1} & \\ s & \frac{m_2-m_N}{2} & \cdots & \frac{m_{N-1}-m_N}{2} & m_N \end{bmatrix} \tag{3.26}$$

où $s = (m_1 - m_2 - \cdots - m_{N-1} + (N-3)m_N)/4$ et les éléments manquants dans Q sont des zéros . On montre alors que la modification conditionnelle peut être mise en œuvre comme suit :

i.	Calculer $l_N = -\lfloor j_N/m_N \rfloor$
	Fixer $j_N' = j_N$
ii.	Pour $i \in \{2, \cdots, N-1\}$:
	– calculer $l_i = -\lfloor (j_i + \frac{m_i - m_N}{2} l_N)/m_i \rfloor$
	– fixer $j_i' = j_i + (m_i - m_N) l_N/2$
iii.	Fixer $j_1' = j_1 + \sum_{i=2}^{N-1} (m_1 - m_i) l_i/2 + s l_N$

Modification conditionnelle pour Λ_{16} : Le développement de l'étape de modification conditionnelle est moins direct dans le cas Λ_{16}. La matrice Q est en effet plus complexe pour Λ_{16}. Celle-ci n'est pas présentée ici, mais elle peut être facilement calculée en fonction de $m_1, \cdots, m_{16}$. La modification conditionnelle pour Λ_{16} est donnée ci-dessous :

i.	Calculer $l_{16} = -\lfloor j_{16}/m_{16} \rfloor$
	Fixer $j'_{16} = j_{16}$
ii.	Pour $i \in \{15, \cdots, 2\}$ (en ordre décroissant) :
	– calculer t_i suivant le tableau 3.2
	– calculer $l_i = -\lfloor (j_i + t_i)/m_i \rfloor$
	– fixer $j'_i = j_i + t_i$
iii.	Calculer t_1 suivant la table 3.2
	Fixer $j'_1 = j_1 + t_1$

TABLEAU 3.2: Valeurs de t_i utilisées dans la modification conditionnelle pour Λ_{16}.

i	t_i
15	$(m_{15} - m_{16})l_{16}$
14	$(m_{14} - m_{16})l_{16}$
13	$(m_{13} - m_{16})l_{16}$
12	$(m_{12} - m_{15})(l_{15} + l_{16})$
11	$[(m_{11} - m_{14})l_{14} + (m_{11} - m_{15})l_{15} + (m_{11} - m_{14} - m_{15} + m_{16})l_{16}]/2$
10	$[(m_{10} - m_{13})l_{13} + (m_{10} - m_{14})l_{14} + (m_{10} - m_{13} - m_{14} + m_{16})l_{16}]/2$
9	$[(m_9 - m_{12})l_{12} + (m_9 - m_{13})l_{13} + (m_9 - m_{13})l_{15} + +(m_9 - m_{12} - m_{13} + m_{16})l_{16}]/2$
8	$[(m_8 - m_{12})l_{12} + (m_8 - m_{14})l_{14} - (m_{12} - m_{15})l_{15} + (m_8 - m_{12} - m_{14} + m_{15})l_{16}]/2$
7	$[(m_7 - m_{13})l_{13} + (m_7 - m_{15})l_{15} + (m_7 - m_{13} - m_{15} + m_{16})l_{16}]/2$
6	$[(m_6 - m_{12})l_{12} + (m_6 - m_{14})l_{14} + (m_6 - m_{12})l_{15} + (m_6 - m_{12} - m_{14} + m_{16})l_{16}]/2$
5	$[(m_5 - m_{13})l_{13} + (m_5 - m_{14})l_{14} + (m_5 - m_{15})l_{15} + (m_5 - m_{13} - m_{14} - m_{15} + m_{16})l_{16}]/2$
4	$[(m_4 - m_{12})l_{12} + (m_4 - m_{13})l_{13} + (m_4 - m_{14})l_{14} + (m_4 - m_{12})l_{15} + (m_4 - m_{12} - m_{13} - m_{14} + m_{16})l_{16}]/2$
3	$[(m_3 - m_{12})l_{12} + (m_3 - m_{13})l_{13} + (m_3 - m_{14})l_{14} - (m_3 - m_{15})l_{15} + (m_3 - m_{12} - m_{13} - m_{14} + m_{15} + m_{16})l_{16}]/2$
2	$[(m_2 - m_{12})l_{12} + (m_2 - m_{13})l_{13} - (m_2 - m_{15})l_{15} + (m_2 - m_{12} - m_{13} + m_{15})l_{16}]/2$
1	$\sum_{i=2}^{11}(m_1 - m_i)l_i/2 + \sum_{i=12}^{16}\alpha_{i,1}l_i$ où $\alpha_{12,1} = (m_1 - m_2 - m_3 - m_4 - m_6 - m_8 - m_9 + 5m_{12})/4$ $\alpha_{13,1} = (-m_2 - m_3 - m_4 - m_5 - m_7 - m_9 - m_{10} + 7m_{13})/4$ $\alpha_{14,1} = (-m_3 - m_4 - m_5 - m_6 - m_8 - m_{10} - m_{11} + 7m_{14})/4$ $\alpha_{15,1} = (-m_4 - m_5 - m_6 - m_7 - m_9 - m_1 1 + 5m_{12} + m_{15})/4$ $\alpha_{16,1} = (m_1 - (\sum_{i=2}^{11} m_i) + 5m_{12} + 7m_{13} + 7m_{14} + m_{15} - 11m_{16})/4$

3.3.4 Complexité

La complexité des algorithmes d'indexation présentés ici est facile à évaluer et surtout indépendante du débit alloué. La mise en œuvre d'un code de Voronoï quasi-ellipsoïdal requiert uniquement de stocker le modulo de Voronoi $\mathbf{m}$, le décalage $\mathbf{a}$, la matrice génératrice $M(\Lambda)$ et son inverse $M(\Lambda)^{-1}$. L'espace mémoire occupé par ces données évolue en $o(N^2)$ avec la dimension N de quantification. La longueur du programme (en langage C par exemple) et la complexité de calcul se résument essentiellement à la recherche du plus proche voisin dans le réseau infini et à l'étape de modification conditionnelle. Elles dépendent du réseau utilisé.

3.4 Algorithmes rapides de recherche du plus proche voisin

La mise en œuvre de la quantification par réseau de points à débit fixe nécessite non seulement des algorithmes de calcul et de décodage d'indice, mais également un algorithme du recherche du plus proche voisin au sein du dictionnaire de quantification. On se restreint ici au cas où le critère de recherche est l'erreur quadratique – la distortion étant mesurée au moyen de l'erreur quadratique moyenne.

Pour un dictionnaire de quantification C donné, la recherche optimale du plus proche voisin d'un vecteur $\mathbf{x} \in \mathbb{R}^N$ dans un code C consiste à minimiser la distance euclidienne dans C par rapport à $\mathbf{x}$. On cherche alors $\hat{\mathbf{x}}$ tel que :

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{y}\in C} \|\mathbf{x} - \mathbf{y}\|^2. \tag{3.27}$$

On s'intéresse ici au cas où C est un code de Voronoï quasi-ellipsoïdal.

Afin d'optimiser la performance de quantification pour la source, le dictionnaire est en fait pris sous la forme $g(C + \delta)$, où $g > 0$ est un facteur d'échelle (ou gain), C un code de Voronoï quasi-ellipsoïdal dans Λ et δ un décalage défini dans $\mathbb{R}^N$. Le décalage δ ne doit pas être confondu avec le décalage de Voronoï $\mathbf{a}$ qui intervient à l'équation 3.5. Il sert à compenser l'asymétrie du code de Voronoï quasi-ellipsoïdal C. Si la source est de moyenne nulle, δ est en général ajusté de sorte que

$C + \delta$ soit quasiment centré.

Une fois le code C défini, le gain g et le décalage δ sont les seuls paramètres (ou degrés de libertés) qui permettent d'optimiser les performances de quantification. On s'intéresse ici au problème de la recherche dans $g(C + \delta)$ avec g, C et δ fixés.

3.4.1 Problème de la recherche dans $g(C + \delta)$: difficulté de la saturation

La recherche exhaustive dans $g(C + \delta)$ peut être effectuée telle qu'indiquée à la figure 3.5. La recherche du plus proche voisin de $\mathbf{x}$ dans $g(C + \delta)$ se ramène à une recherche dans C pour la cible de quantification $\mathbf{x}' = \frac{1}{g}\mathbf{x} - \delta$:

$$\|\mathbf{x} - g(\mathbf{y} + \delta)\|^2 = g^2 \|(\frac{1}{g}\mathbf{x} - \delta) - \mathbf{y}\|^2 = g^2 \|\mathbf{x}' - \mathbf{y}\|^2. \tag{3.28}$$

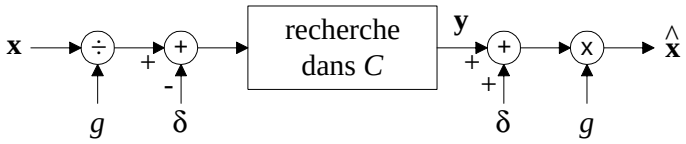


Figure 3.5: Recherche optimale dans le dictionnaire de quantification quasi-ellipsoïdal $g(C + \delta)$.

En pratique, cette recherche exhaustive n'est pas réalisable ou intéressante pour des raisons de complexité. La recherche du plus proche voisin de $\mathbf{x}$ dans $g(C + \delta)$ est plutôt réalisée en 3 étapes :

1. Recherche du plus proche voisin $\mathbf{y}$ de $\mathbf{x}' = \frac{1}{g}\mathbf{x} - \delta$ dans Λ ;
2. Détection de surcharge, qui revient à vérifier si $\mathbf{y} \in C$;
3. En cas de surcharge (si $\mathbf{y} \notin C$), saturation de $\mathbf{x}'$ dans C.

Ces étapes sont examinées séparément dans les paragraphes suivants.

Recherche du plus proche voisin dans le réseau infini Λ

La recherche du plus proche voisin dans le réseau infini Λ suivant l'erreur quadratique est indépendante de la troncature du réseau. Cette opération est décrite à la section 2.3.

Détection de surcharge

Dans le cas du codage de Voronoï quasi-ellipsoïdal, la détection (binaire) de surcharge peut être facilement réalisée à l'aide des algorithmes d'indexation $\mathbf{y} \rightarrow \mathbf{k}$ et $\mathbf{k} \rightarrow \mathbf{y}$. En effet, il y a surcharge si après avoir calculé $\mathbf{y} \rightarrow \mathbf{k} \rightarrow \mathbf{z}$, $\mathbf{z}$ et $\mathbf{y}$ diffèrent. Cette détection se ramène à un décodage dans Λ.

Si le décalage $\mathbf{a}$ est admissible, une autre solution consiste simplement à trouver le plus proche voisin $\mathbf{z}$ de $\mathbf{div}(\mathbf{y} - \mathbf{a}, \mathbf{m})$ dans Λ, puis vérifier que $\mathbf{z}$ est nul. Dans ce cas, la vérification requiert également un décodage de Λ.

Pour éviter ce décodage de réseau, on peut vérifier que $\mathbf{y}$ est à l'interieur d'un ellipsoïde centré par exemple en $\mathbf{a}$ d'équation $(\mathbf{x} - \mathbf{a})\Sigma^{-1}(\mathbf{x} - \mathbf{a})^t \leq \rho^2$ où $\Sigma = diag(m_1^2, \cdots, m_N^2)$ et ρ est le rayon d'empilement de Λ. Mais ce test est approximatif car certains mots de code ne sont pas dans l'ellipsoïde.

Saturation

La saturation est délicate à résoudre dans le cas du codage de Voronoï. La troncature de Voronoï rend en effet quasiment impossible la réalisation d'une saturation optimale, car l'ensemble des mots de code proches de la frontière de saturation ne possède pas de structure évidente et facilement exploitable.

Plusieurs stratégies ont été proposées pour s'attaquer au problème de la surcharge dans un dictionnaire construit par troncature d'un réseau de points [17] [18, p.2211]. Le vecteur à quantifier peut être projeté sur la région de troncature – éventuellement dilatée. Puis on applique un décodage

dans Λ [19, 10, 20]. Il est en effet probable que le plus proche voisin de la projection soit un mot de code. La recherche peut aussi être effectuée dans une région de taille définie autour de $\mathbf{x}'$ ou de sa projection [21, 18]. Pour certaines troncatures, la saturation peut être réalisée de façon optimale. C'est le cas de la troncature par un cube où la recherche peut s'appuyer sur un treillis et un calcul de métriques scalaires [22]. C'est aussi le cas de la troncature spécifiée par leaders où la saturation peut être effectuée en appliquant des algorithmes rapides de quantification vectorielle algébrique sphérique à partir d'un sous-ensemble des leaders définissant le code C.

Le problème de la saturation dans un code de Voronoï est relativement peu exploré [2, 3, 4]. Les algorithmes de saturation de [2, 3] sont sous-optimaux. Dans [2], la saturation repose sur une réduction itérative géométrique de [2]. La complexité de cette réduction itérative n'est en théorie pas bornée ; elle peut cependant être réduite en appliquant une dichotomie à progression discrète géométrique ou arithmétique. Dans [3] la saturation est effectuée par projection orthogonale sur la région de troncature et réduction itérative arithmétique. Cette projection est illustrée à la figure 3.6. La projection $\mathbf{h}$ de $\mathbf{x}'$ permet de trouver un point $\mathbf{y}$ dans C proche de $\mathbf{x}'$. Néanmoins $\mathbf{y}$ n'est pas ici le plus proche voisin de $\mathbf{x}'$ dans C.

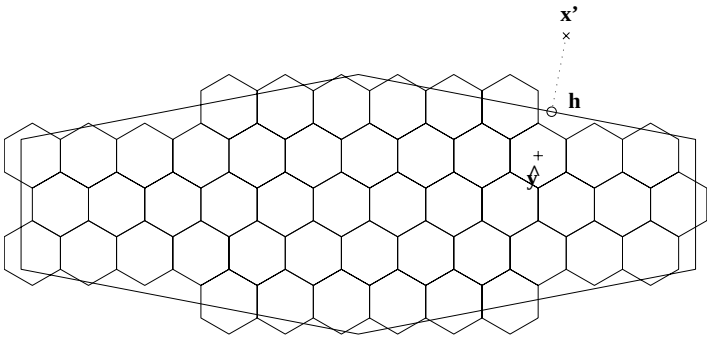


Figure 3.6: Exemple de projection en cas de surcharge dans un code quasi-ellipsoïdal (cas de A_2).

La saturation par réduction itérative (avec ou sans projection) est par nature sous-optimale. Une approche optimale consiste à spécifier la frontière de saturation en terme d'orbites ellipsoïdales [4]. Elle requiert cependant une énumération compliquée des orbites qui n'est envisageable que pour des dimensions de quantification faibles ($N \leq 8$) et de bas débits. De plus, elle convient mieux aux vrais

codes algébriques ellipsoïdaux de [10, 11, 12]. Pour ces raisons, elle n'est donc pas étudiée ici. Une approche alternative consiste à projeter $\mathbf{x}'$ sur la région de troncature, puis à chercher autour de la projection tous les points de Λ inclus dans une sphère de rayon fixé [18]. Seuls les points de Λ qui sont des mots de code dans C sont retenus ; le plus proche de $\mathbf{x}'$ est sélectionné. Néanmoins cet algorithme alternatif est a priori très complexe.

3.4.2 Algorithme rapide de recherche dans $g(C+\delta)$

On présente ici un algorithme de recherche du plus proche voisin dans $g(C+\delta)$ par projection et réduction. Il intègre plusieurs nouveautés par rapport à [3, 2] :

- La saturation fait appel à une projection orthogonale ou radiale sur $V(\Lambda')$, ou à une projection sur un ellipsoïde dont la forme s'approche de $V(\Lambda')$;
- La projection sur $V(\Lambda')$ est réalisée via une recherche du vecteur pertinent de $V(\Lambda)$;
- Une dichotomie est utilisée pour trouver rapidement un facteur de réduction adéquat ;
- Le décalage δ est pris en compte.

Algorithme détaillé

L'algorithme de recherche dans $g(C+\delta)$ est illustré à la figure 3.7. Il présuppose que le code $C+\delta$ est approximativement centré. En pratique, on impose que $\delta \in V(\Lambda)$.

Les étapes de la recherche du plus proche voisin sont données à la figure 3.8. En cas de surcharge, le vecteur $\mathbf{w}$ est tout d'abord projeté. La projection $\mathbf{h}$ de $\mathbf{w}$ est réduite automatiquement sous la forme $\alpha\mathbf{h}$ jusqu'à ce qu'il n'y ait plus de surcharge. Pour trouver rapidement un facteur α adéquat, une dichotomie est utilisée. Le facteur α est cherché dans un intervalle $[\alpha_1, \alpha_2]$ (avec $\alpha_2 > \alpha_1$) qui diminue, jusqu'à ce que $|\alpha_1 - \alpha_2| \leq \epsilon$ (avec $0 < \epsilon < 1$).

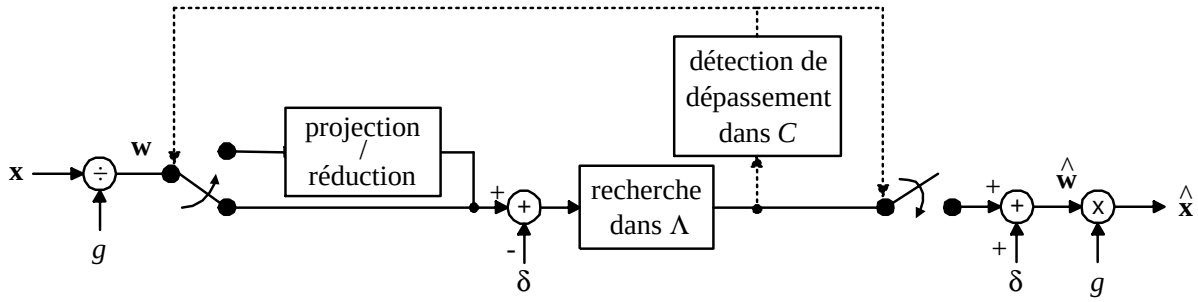


Figure 3.7: Recherche sous-optimale dans un code quasi-ellipsoïdal $g(C+\delta)$.

1.	Calculer $\mathbf{w} = \frac{1}{g}\mathbf{x}$.
2.	Trouver le plus proche voisin $\mathbf{y}$ de $\mathbf{w} - \delta$ dans Λ
3.	Si $\mathbf{y}$ est un mot de code (i.e. $\mathbf{y} \in C$), aller à l'étape 9.
4.	Calculer la projection $\mathbf{h}$ de $\mathbf{w}$ sur $V(\Lambda')$ ou sur un ellipsoïde. Si $\mathbf{h} = \mathbf{w}$, aller à l'étape 7.
5.	Trouver le plus proche voisin $\mathbf{y}$ de $\mathbf{h} - \delta$ dans Λ
6.	Si $\mathbf{y}$ est un mot de code (i.e. $\mathbf{y} \in C$), aller à l'étape 9.
7.	Fixer : $\alpha_1 = 0$, $\alpha_2 = 1$, $\mathbf{y} = (0, \cdots, 0)$
8.	Tant que $\lvert\alpha_1 - \alpha_2\rvert > \epsilon$:
	(i) calculer $\alpha = (\alpha_1 + \alpha_2)/2$
	(ii) trouver le plus proche voisin $\mathbf{t}$ de $\alpha\mathbf{h} - \delta$ dans Λ
	(iii) si $\mathbf{t}$ est un mot de code (i.e. $\mathbf{t} \in C$), fixer $\alpha_1 = \alpha$ et $\mathbf{y} = \mathbf{t}$
	sinon poser $\alpha_2 = \alpha$
9.	Calculer $\hat{\mathbf{x}} = g(\mathbf{y} + \delta)$.

Figure 3.8: Algorithme de recherche par projection et réduction : $\mathbf{x} \to \hat{\mathbf{x}} \in g(C+\delta)$

Des algorithmes rapides de projection sur $V(\Lambda')$ et sur un ellipsoïde sont présentés au paragraphe suivant.

Algorithmes de projection

Pour simplifier la projection, on suppose que la source $\mathbf{x}$ est de moyenne nulle, que le code $C+\delta$ est approximativement centré, et que le décalage $\mathbf{a}$ (qui intervient dans la définition de C) est nul. Dans ces conditions, la région de projection (quasi-)optimale est centrée. On considère plusieurs types de projections : des projections orthogonale et radiale sur la région de Voronoï $V(\Lambda')$, ainsi

qu'une projection sur un ellipsoïde centré.

Projection sur la région de Voronoï $V(\Lambda')$

La projection sur $V(\Lambda')$ nécessite de connaître la structure géométrique de $V(\Lambda')$. En faible dimension (pour $N \leq 4$), il est envisageable d'utiliser les sommets de $V(\Lambda')$, c'est-à-dire les trous de Λ' autour de l'origine [23]. Dans le cas général, on peut identifier la *face pertinente* de $V(\Lambda')$ correspondant au point à projeter en cherchant le *vecteur pertinent* le plus proche. Cette dernière solution est utilisée ici.

Identification de la face pertinente par recherche du vecteur pertinent le plus proche :

Pour être en mesure de projeter $\mathbf{x}$ sur $V(\Lambda')$, il faut connaître la face pertinente de $V(\Lambda')$ associée à $\mathbf{x}$. Celle-ci peut être trouvée en identifiant la face pertinente de $V(\Lambda)$ correspondant à $\mathbf{t} = \mathbf{div}(\frac{1}{g}\mathbf{x}, \mathbf{m}) = \mathbf{div}(\mathbf{w}, \mathbf{m})$. On se réduit donc ici à étudier la géométrie de $V(\Lambda)$. On trouve certaines généralités sur $V(\Lambda)$ au chapitre 2.

La région de Voronoï $V(\Lambda)$ est un polytope convexe de volume $det(M(\Lambda))$ et ayant pour sommets les trous de Λ. Ses faces de dimension $N-1$ sont définies par les vecteurs pertinents de Λ [14] ; chacune des faces de $V(\Lambda)$ fait partie d'un hyperplan médiateur affine de dimension $N-1$ entre l'origine et un vecteur pertinent $\mathbf{r}$ de Λ.

D'après [14, p.461], dans le cas de A_2, D_N et $2D_8^+$, les vecteurs pertinents de Λ sont précisément les vecteurs de norme minimale de Λ et $V(\Lambda)$ compte autant de faces de dimension $N-1$ que le nombre de contacts τ de Λ. La face de $V(\Lambda')$ associée à $\mathbf{x}$ peut dans ce cas être identifiée en cherchant le vecteur $\mathbf{r}$ de Λ le plus proche – au sens de l'erreur quadratique – de $\mathbf{t} = \mathbf{div}(\mathbf{w}, \mathbf{m})$ parmi τ vecteurs de norme minimale. Puisque ces τ vecteurs pertinents sont sur une même couche sphérique de Λ, la recherche rapide de $\mathbf{r}$ peut être réalisée par des techniques de quantification vectorielle algébrique par orbite sphérique [24] à base de leaders.

En général, les vecteurs pertinents de Λ comprennent les τ vecteurs de norme minimale, ainsi que des vecteurs des couches sphériques consécutives. On peut les calculer en appliquant les algorithmes

de [21, 18]. Les mêmes techniques de quantification vectorielle algébrique par orbite sphérique [24] peuvent être appliquées pour chercher $\mathbf{r}$ efficacement.

Les vecteurs pertinents pour les réseaux A_2, D_N ($N \geq 2$), $2D_N^+$ (N pair ≥ 4) et Λ_{16} utilisés ici sont énumérés au tableau 3.3. Puisque $2D_4^+ \cong 2\mathbb{Z}^4$, le nombre de vecteurs pertinents pour $2D_4^+$ est $2N = 8$. Le nombre de vecteurs pertinents pour Λ_{16} est tiré de [25].

TABLEAU 3.3: Paramètres pertinents pour l'identification d'une face de la région de Voronoï.

Λ	τ	Vecteurs pertinents de Λ	
		Nombre	Forme
A_2	6	6	$(\pm 1, 0)$, $(\pm\frac{1}{2}, \pm\frac{\sqrt{3}}{2})$
D_N ($N \geq 2$)	$2N(N-1)$	$2N(N-1)$	permutations de $((\pm 1)^2, 0^{N-2})$
$2D_N^+$ (N pair ≥ 4)	2^{N-1} ($N \leq 7$) 240 ($N = 8$) $2N(N-1)$ ($N \geq 9$)	8 ($N = 4$) $2^{N-1} + 2N(N-1)$ ($N > 4$)	$(\pm 1^N)$ avec parité de signes négatifs et si $N > 4$ permutations de $(\pm 2^2, 0^{N-2})$
Λ_{16}	4320	65760	2 premières sphères de Λ_{16}

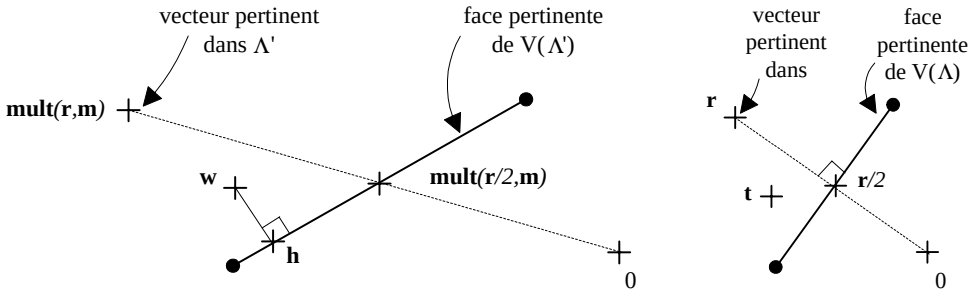
Projection orthogonale :

La projection orthogonale consiste à déplacer $\mathbf{t}$ suivant le vecteur normal $\mathbf{s}$ de la face pertinente de $V(\Lambda')$ de sorte que $\mathbf{h} = \mathbf{t} - \beta\mathbf{s}$ soit sur la face, tel qu'illustré à la figure 3.9 (a). Pour réaliser cette projection, on peut d'abord ramener $\mathbf{w}$ dans le domaine de Λ, comme illustré à la figure 3.9 (b). Le vecteur pertinent $\mathbf{r}$ de Λ associé à $\mathbf{t} = \mathbf{div}(\mathbf{x}, \mathbf{m})$ est ainsi calculé. Alors en revenant à Λ', le vecteur normal $\mathbf{s}$ est donné par $\mathbf{s} = \mathbf{div}(\mathbf{r}, \mathbf{m})$. Le facteur β peut être calculé en observant que $(\mathbf{t} - \mathbf{h})(\mathbf{h} - \mathbf{mult}(\mathbf{r}/2, \mathbf{m}))^t = 0$, soit $\mathbf{s}(-\beta\mathbf{s} + \mathbf{t} - \mathbf{mult}(\mathbf{r}/2, \mathbf{m}))^t = 0$.

La projection orthogonale garantit uniquement que le point projeté est sur l'hyperplan contenant la face pertinente de $V(\Lambda')$. Un mécanisme supplémentaire est donc nécessaire pour s'assurer que la projection orthogonale est sur $V(\Lambda')$. Ce mécanisme se réduit ici à remplacer la projection orthogonale par une projection radiale. L'algorithme est détaillé à la figure 3.10.

Projection radiale :

La projection radiale consiste à utiliser un facteur d'échelle α à $\mathbf{w}$ de sorte que $\mathbf{h} = \alpha\mathbf{w}$ appar-



(a) par rapport à Λ' (b) par rapport à Λ

Figure 3.9: Projection orthogonale sur la face pertinente de $V(\Lambda')$: $\mathbf{h} = \mathbf{x} - \beta\mathbf{s}$ où $\mathbf{s}$ est le vecteur normal à la face pertinente.

(i)	Identifier la face de $V(\Lambda)$ associée à $\mathbf{t} = \mathbf{div}(\frac{1}{g}\mathbf{x}, \mathbf{m}) = \mathbf{div}(\mathbf{w}, \mathbf{m})$ en cherchant le vecteur pertinent $\mathbf{r}$ le plus proche de $\mathbf{t}$ parmi les vecteurs pré-définis de Λ
(ii)	Calculer $\alpha = \frac{1}{2}\mathbf{r}\mathbf{r}^t/\mathbf{t}\mathbf{r}^t$. Si $\alpha \geq 1$, poser $\mathbf{h} = \mathbf{w}$. Fin.
(iii)	Sinon calculer le vecteur normal $\mathbf{s}$ à la face de $V(\Lambda')$: $\mathbf{s} = \mathbf{div}(\mathbf{r}, \mathbf{m})$.
(iv)	Projeter $\mathbf{x}$ orthogonalement sur $\mathbf{h}$ qui est sur l'hyperplan contenant la face appropriée de $V(\Lambda')$, i.e. calculer $\mathbf{h} = \mathbf{w} - \beta\mathbf{s}$ avec $\beta = \mathbf{s}(\mathbf{w} - \mathbf{mult}(\mathbf{r}, \mathbf{m})/2)^t/\mathbf{s}\mathbf{s}^t$.
(v)	Vérifier que les vecteurs pertinents de Λ associées à $\mathbf{t}$ et $\mathbf{div}(\mathbf{h}, \mathbf{m})$ sont identiques. Dans le cas contraire, la projection $\mathbf{h}$ est remplacée par une projection radiale, soit $\mathbf{h} = \alpha\mathbf{w}$.

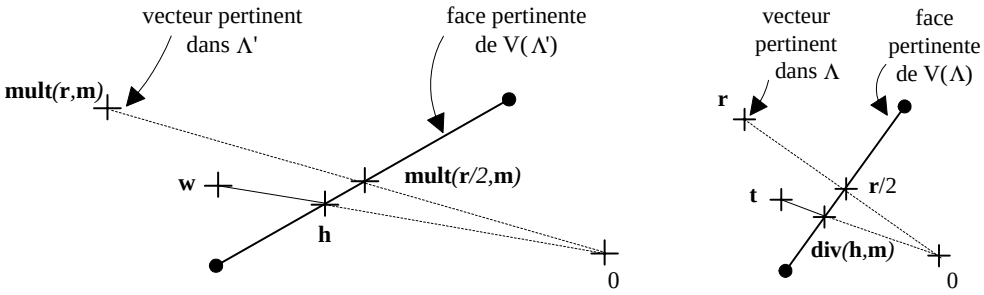
Figure 3.10: Algorithme de projection orthogonale sur $V(\Lambda')$.

tienne à la face pertinente de $V(\Lambda')$, comme indiqué à la figure 3.11 (a). Le facteur α adéquat peut être calculé en ramenant $\mathbf{w}$ dans le domaine de Λ au lieu de Λ', comme illustré à la figure 3.11 (b). La face pertinente de $V(\Lambda)$ est orthogonale au vecteur pertinent $\mathbf{r}$, si bien que $(\alpha\mathbf{t} - \mathbf{r}/2)\mathbf{r}^t = 0$.

Un algorithme de projection radiale est présenté à la figure 3.12. Par définition, le résultat $\mathbf{h} = \alpha\mathbf{w}$ vérifie $\rho^2 \leq \mathbf{h}\Sigma^{-1}\mathbf{h}^t \leq R^2$ où $\Sigma = diag(m_1^2, \cdots, m_N^2)$, car la région $V(\Lambda)$ est inscrite (resp. circonscrite) dans une sphère de rayon ρ (resp. R).

Projection sur un ellipsoïde

Parce que la région de troncature $V(\Lambda')$ est un polytope en général à géométrie complexe en di-



(a) par rapport à Λ' (b) par rapport à Λ

Figure 3.11: Projection radiale sur la face pertinente de $V(\Lambda')$: $\mathbf{h} = \alpha\mathbf{x}$.

(i)	Chercher le vecteur pertinent $\mathbf{r}$ le plus proche de $\mathbf{t} = \mathbf{div}(\mathbf{w}, \mathbf{m}) = \mathbf{div}(\frac{1}{g}\mathbf{x}, \mathbf{m})$ parmi les vecteurs pré-définis de Λ
(ii)	Calculer $\alpha = \frac{1}{2}\mathbf{r}\mathbf{r}^t / \mathbf{t}\mathbf{r}^t$.
(iii)	Si $\alpha \geq 1$, poser $\mathbf{h} = \mathbf{x}$; autrement calculer $\mathbf{h} = \alpha\mathbf{w}$.

Figure 3.12: Algorithme de projection radiale sur $V(\Lambda')$.

mension N élevée, la projection sur $V(\Lambda')$ est une opération relativement compliquée. Cette projection peut en fait être largement simplifiée en remplaçant ce polytope par un ellipsoïde. L'algorithme correspondant est décrit à la figure 3.13.

(i)	Calculer $\mathbf{t} = \mathbf{div}(\mathbf{w}, \mathbf{m}) = \mathbf{div}(\frac{1}{g}\mathbf{x}, \mathbf{m})$
(ii)	Si $\|\mathbf{t}\| \leq r$, poser $\mathbf{h} = \mathbf{w}$; autrement calculer $\mathbf{h} = \frac{r}{\|\mathbf{t}\|}\mathbf{w}$

Figure 3.13: Algorithme de projection sur un ellipsoïde.

L'ellipsoïde est centré et paramétré par un rayon r fixé par exemple entre ρ (le rayon d'empilement de Λ) et R (le rayon de couverture de Λ). Les vecteurs déjà à l'intérieur de l'ellipsoïde sont inchangés. La projection conditionnelle $\mathbf{h}$ appartient à l'ellipsoïde d'équation $\mathbf{h}\Sigma^{-1}\mathbf{h}^t \leq r^2$ où $\Sigma = diag(m_1^2, \cdots, m_N^2)$. En d'autres termes, après projection, la distance de Mahalanobis entre $\mathbf{x}$ et l'origine est inférieure ou égale à r^2. On prend ici $r = \rho$.

3.5 Optimisation pour la source des paramètres du codage de Voronoï

Les codes de Voronoï quasi-ellipsoïdaux $g(C+\delta)$ sont paramétrés pour un réseau Λ donné par $\mathbf{m}$, $\mathbf{a}$, g et δ. Plusieurs contraintes sont définies sur ces paramètres :

- Le vecteur $\mathbf{m}$ doit être un modulo de Voronoï admissible de Λ. De plus, en pratique, la taille du code C doit satisfaire à une certaine allocation de bits R par dimension ; le vecteur $\mathbf{m} = (m_1, \cdots, m_N)$ doit donc aussi vérifier $\prod_{i=1}^{N} m_i \leq 2^{NR}$.
- Par ailleurs, le décalage $\mathbf{a}$ doit être fixé de sorte qu'aucun point de Λ ne soit sur la frontière de $V(\Lambda') + \mathbf{a}$. En réalité, cette dernière contrainte peut être levée, à condition que l'algorithme de recherche du plus proche voisin dans Λ' soit bien conditionné.

Étant données ces conditions, on s'intéresse ici à optimiser $\mathbf{m}$, $\mathbf{a}$, g et δ pour une source gaussienne vectorielle $\mathbf{x}$. Sans perte de généralité, on suppose que $\mathbf{x}$ est à moyenne nulle, à composantes indépendantes et identiquement distribuées et de matrice de covariance $diag(\sigma_1^2, \cdots, \sigma_N^2)$.

A priori, les paramètres $\mathbf{m}$, $\mathbf{a}$, g et δ doivent être optimisés conjointement pour la source. De plus, leurs valeurs optimales sont en théorie fonction de l'algorithme de recherche (optimal ou sous-optimal) du plus proche voisin employé, car la forme des régions de décodage dépend de cet algorithme. Pour éviter des développements trop complexes, on suppose que l'optimisation de ces paramètres peut être réalisée séparément et indépendamment de l'algorithme de recherche du plus proche voisin.

3.5.1 Optimisation du modulo de Voronoï m

La contrainte d'admissibilité sur $\mathbf{m}$ étant dépendante de Λ, l'optimisation de $\mathbf{m}$ dépend de Λ.

Puisque le vecteur $\mathbf{m}$ donne en quelque sorte le nombre de niveaux de quantification sur chaque composante, les techniques d'allocation des bits peuvent être utilisées. L'allocation des bits optimale au sens de la théorie de la distorsion à haut débit pour une source gaussienne $\mathbf{x}$ est par exemple

décrite dans [17, p. 231]. Si la source $\mathbf{x}$ stationnaire est codée par transformation de Karhunen-Loève et quantification scalaire, le nombre de niveaux de quantification nécessaire à la quantification de la i-ème composante (après rotation la ramenant sur ses axes principaux) doit être proportionnel à l'écart-type σ_i du i-ème coefficient de la transformée de Karhunen-Loève de $\mathbf{x}$. En terme de débit, le nombre de bits alloués à la i-ème composante doit être approximativement

$$\theta + 1/2 \log_2 \frac{\sigma_i}{(\prod_{i=1}^{N} \sigma_i)^{1/N}}, \tag{3.29}$$

où θ est un nombre fixe de bits [26].

Ce résultat classique s'interprète aussi géométriquement dans le cas de la quantification par réseau de points [17, p .474]. En effet, pour optimiser les performances d'un code algébrique, la forme de la troncature du code doit ressembler aux contours d'équiprobabilité de la source – ces contours sont ici des ellipsoïdes d'équation $\mathbf{x}\Sigma^{-1}\mathbf{x}^t = r^2$ avec $\Sigma = diag(\sigma_1^2, \cdots, \sigma_N^2)$ et r est un rayon ≥ 0. Ce résultat correspond aux idées de [27, 10] sur le codage de source géométrique.

On trouve dans tous les cas que, pour $1 \leq i \leq N$, la composante m_i du modulo de Voronoï $\mathbf{m}$ doit être proportionnelle à σ_i, le facteur de proportionalité étant identique pour chaque composante. Pour un budget de NR bits par vecteur, on trouve alors que $\mathbf{m}$ doit être proche de $\mathbf{m}^* = \beta(\sigma_1, \cdots, \sigma_N)$ avec

$$\beta = \left(\frac{2^{NR}}{\prod_{i=1}^{N} \sigma_i} \right)^{1/N} = \frac{2^R}{\left(\prod_{i=1}^{N} \sigma_i \right)^{1/N}}. \tag{3.30}$$

Cependant $\mathbf{m}^*$ n'est pas en général un vecteur entier. Il est toutefois envisageable d'optimiser $\mathbf{m}$ en cherchant tous les vecteurs $\mathbf{m}$ admissibles proches de $\mathbf{m}^*$ et tels que $\prod_i m_i \leq 2^B$.

Pour développer une optimisation de $\mathbf{m}$ entière, on s'inspire plutôt ici de l'algorithme glouton [17]. Cet algorithme réalise une optimisation de type *reverse waterfilling*. Une version de cet algorithme dans le cas où l'ensemble d'admissibilité est $\mathbb{Z} \setminus \{0, 1\}^N$ est détaillée à la figure 3.14..

Cet algorithme peut être facilement adapté à d'autres ensembles d'admissibilité que $(\mathbb{Z} \setminus \{0, 1\})^N$. En général l'ensemble d'admissibilité de $\mathbf{m}$ est $(\mathbb{Z} \setminus \{0, 1\})^N \cap A$ où A est le réseau régulier défini au paragraphe 3.2.2. Au lieu de modifier $\mathbf{m}$ par incrément de ± 1 sur la $i$ème composante, $\mathbf{m}$ doit être

1.	Initialiser $\mathbf{m} = (2, \cdots, 2)$ et $\mathbf{s} = (\sigma_1^2, \cdots, \sigma_N^2)/4$
2.	Trouver l'indice i de la composante maximale de $\mathbf{s}$: $i = \arg \max_{1 \leq j \leq N} s_j$
3.	Fixer $m_i := m_i + 1$
4.	Si $(\prod_{i=1}^{N} m_i) > 2^B$, fixer $m_i := m_i - 1$. Aller à l'étape 5.
5.	Mettre à jour s_i : $s_i = (\sigma_i / m_i)^2$. Aller à l'étape 1.
5.	Essayer (itérativement) d'incrémenter les composantes de $\mathbf{m}$ (en allant de i_1 à i_N avec $s_{i_1} \geq \cdots \geq s_{i_N}$) – jusqu'à ce qu'il soit impossible d'incrémenter les composantes de $\mathbf{m}$, c'est-à-dire quand les positions $i = 1, \cdots, N$ sont toutes bloquées.

Figure 3.14: Algorithme glouton (optimisation du modulo de Voronoï).

translaté par un vecteur de A à chaque itération – la translation optimale étant celle qui minimise $\|\mathbf{s}\|^2$ pour respecter le principe de l'allocation des bits optimale au sens de l'erreur quadratique (*reverse waterfilling*).

On traite ici quelques cas particuliers. Pour $\Lambda = D_N$, un incrément de $+2$ sur m_i (au lieu de $+1$) doit être utilisé. Pour $\Lambda = 2D_N^+$, à chaque itération, on doit ajouter à $\mathbf{m}$ un vecteur minimal de $2D_N^+$ en fonction de $\mathbf{s}$. Pour simplifier, on peut utiliser l'algorithme glouton avec un incrément de $+2$, et s'assurer que le résultat $\mathbf{m}$ est tel que $m_1 + \cdots + m_N$ est un multiple de 4 – dans le cas contraire, on doit fixer $m_i := m_i - 2$ avec i approprié. Alternativement, on peut ajouter uniquement les points pairs de la forme $(\pm 2, \pm 2, 0, \cdots, 0)$ après avoir localisé les 2 plus grandes valeurs dans $\mathbf{s}$.

Exemple :
On prend $N = 8$, $B = 20$ bits et $\Sigma = diag(18^2, 13^2, 9^2, 6^2, 4^2, 3^2, 2^2, 2^2)$. On trouve: $\mathbf{m}^* \approx (19.2749, 13.9207, 9.6374, 6.4250, 4.2833, 3.2125, 2.1417, 2.1417)$.

Pour $\Lambda = D_8$, les modulo de Voronoï admissibles sont des points de $2D_8^*$. L'algorithme glouton adapté avec des pas de $+2$ à chaque itérations sur $\mathbf{m}$ donne : $\mathbf{m} = (18, 14, 10, 6, 4, 4, 2, 2)$.

Pour $\Lambda = RE_8 = 2D_8^+$, les modulo de Voronoï admissibles sont des points de $2D_8^+$. L'algorithme glouton avec une initialisation à $\mathbf{m} = (2, \cdots, 2)$ et des translations sur $\mathbf{m}$ de la forme $(+2, +2, 0, \cdots, 0)$ où les $+2$ sont positionnés suivant les deux plus grands éléments courants de $\mathbf{s}$ donne : $\mathbf{m} = (18, 14, 10, 6, 4, 4, 2, 2)$.

3.5.2 Optimisation de a

L'optimisation de $\mathbf{a}$ est problématique car aucun lien évident n'existe entre $\mathbf{a}$ et la distorsion. On peut appliquer une approche par essai-erreur ou la procédure itérative de [1] (adaptée au codage de Voronoï quasi-ellipsoïdal). Dans ce dernier cas, on impose que le vecteur $\mathbf{a}$ soit le barycentre de C. Le décalage $\mathbf{a}$ est initialisé à un vecteur admissible ; ensuite, il est remplacé à chaque itération par le barycentre de C jusqu'à convergence de $\mathbf{a}$. Cependant, cette optimisation ne vise pas directement à minimiser la distorsion.

On prend ici pour simplifier $\mathbf{a} = (0, \cdots, 0)$. Ce décalage nul n'est pas admissible. Néanmoins en pratique la contrainte d'admissibilité sur $\mathbf{a}$ peut être levée, si la détection de dépassement dans C est effectuée comme au paragraphe 3.4.1 (en calculant $\mathbf{y} \rightarrow \mathbf{k} \rightarrow \mathbf{z}$ puis en vérifiant que $\mathbf{y}$ et $\mathbf{z}$ coincident).

3.5.3 Optimisations de g et δ

Le décalage δ et le gain g doivent être idéalement optimisés conjointement. En pratique, un algorithme de type séquentiel itératif peut être développé à partir d'une base d'apprentissage de la source $\mathbf{x}$, où le gain g est optimisé par recherche par gradient empirique pour δ fixe, tandis que le décalage δ est optimisé pour g fixe. Ces deux opérations peuvent être repétées jusqu'à convergence de la distorsion.

On fixe simplement ici le décalage δ à l'opposé du centroïde de C, le gain g étant optimisé par la méthode de la descente du gradient. En pratique, cette approche est légèrement sous-optimale par rapport à une optimisation complète de δ. Pour accélerer l'optimisation de g, la recherche par gradient peut recourir à un modèle de distorsion au lieu de calculer la distorsion réelle ; un tel modèle est présenté au paragraphe 3.6.5.

3.6 Performances et complexité de la recherche du plus proche voisin : exemple basé sur D_8 et $RE_8 = 2D_8^+$

On ne considère ici qu'un exemple de quantification en dimension $N = 8$. Les codes utilisés sont dérivés de deux réseaux de points en dimension 8 : le réseau en damier D_8 et une version tournée du réseau de Gosset RE_8. On rappelle ici que [14]

$$D_8 = \left\{(x_1, \cdots, x_8) \in \mathbb{Z}^8 | x_1 + \cdots + x_8 \text{ est pair}\right\}. \tag{3.31}$$

De plus [22],

$$RE_8 = 2D_8^+ = 2D_8 \cup \{2D_8 + (1, \cdots, 1)\}. \tag{3.32}$$

Les propriétés des ces réseaux utiles à la recherche du plus proche voisin sont rappelées au tableau 3.4.

TABLEAU 3.4: Propriétés pertinentes des réseaux D_8 et RE_8.

	D_8	RE_8
Rayon d'empilement (ρ)	$1/\sqrt{2}$	$\sqrt{2}$
Rayon de couverture (R)	$\sqrt{2}$	2
Nombre de points de contacts (τ)	112	240
Forme des vecteurs de norme minimale	$(\pm 1^2, 0^6)$	$(\pm 2^2, 0^6)$ $(\pm 1^8)$ avec un nombre pair de signes négatifs

3.6.1 Conditions expérimentales

- Caractéristique de la source : La source $\mathbf{x}$ à quantifier est gaussienne en dimension 8, centrée et de matrice de covariance $diag(18^2, 13^2, 9^2, 6^2, 4^2, 3^2, 2^2, 2^2)$. Il s'agit de la même source que celle utilisée dans [2, chap 1]. La méthode de Box-Muller [28] est employée pour générer des nombres aléatoires distribués suivant la loi normale $\mathcal{N}(0, 1)$. Pour l'évaluation des performances, la source $\mathbf{x}$ est simulée en générant des bases de test comprenant 96,768 vecteurs pour la recherche exhaustive et 9,676,800 vecteurs pour les recherches par projection et réduction.

- Paramètres des codes de Voronoï quasi-ellipsoïdaux : La source est représentée par quantification de type gain-forme avec un dictionnaire $g(C+\delta)$. Mais contrairement à [13], le dictionnaire n'est pas de norme unité et la forme n'est pas quantifiée avant le gain.

  Les mêmes paramètres $\mathbf{m}$, $\mathbf{a}$, g et δ sont utilisés pour tous les types de recherche dans le dictionnaire de quantification $g(C+\delta)$, afin de comparer les algorithmes associés sur une base équitable.

  Le code de Voronoï C dans $\Lambda = D_8$ ou $RE_8 = 2D_8^+$ est spécifié par $\mathbf{m}$ et $\mathbf{a}$. On fixe $\mathbf{m} = (18, 14, 10, 6, 4, 4, 2, 2)$ dans les deux cas. On peut vérifier que ce modulo de Voronoï, le même que dans [2], est admissible pour D_8 et RE_8. Les codes comprennent donc $967,680 \approx 2^{19.98}$ mots de code pour un débit par dimension d'approximativement 2.4855 bits. Le décalage $\mathbf{a}$ est nul pour D_8 et pour RE_8. Le décalage δ est fixé à l'opposé du centroïde de C.

- Paramètres des algorithmes de recherche dans $g(C+\delta)$: Les algorithmes de recherche dans $g(C+\delta)$ comparés ici sont spécifiés au tableau 3.5. Le paramètre ϵ de la recherche par projection et réduction (avec dichotomie) est fixé à $\epsilon = 0.01$. La recherche exhaustive est réalisable pour le débit visé. Elle est mise en œuvre en 2 étapes pour réduire sa complexité moyenne. On cherche d'abord le plus proche voisin dans Λ. Puis, une recherche similaire à une quantification vectorielle non structurée est appliquée, si ce plus proche voisin n'est pas un mot de code.

TABLEAU 3.5: Algorithmes de recherche testés.

Référence	Algorithme de recherche
OPT	recherche exhaustive
ORTH	projection orthogonale sur $V(\Lambda')$, réduction arithmétique
RAD	projection radiale sur $V(\Lambda')$, réduction arithmétique
ELLIP	projection sur ellipsoïde centrée ($r = \rho$), réduction arithmétique

3.6.2 Calcul de la borne débit-distorsion

La source gaussienne donnée au paragraphe 3.6.1 a une matrice de covariance diagonale. Dans ce cas les performances optimales de quantification (i.e. la limite de Shannon) peuvent être calculées directement suivant le principe de remplissage inverse des eaux (*reverse waterfilling*), sans transformation de Karhunen-Loève. Pour un budget de bits de 20 bits par vecteur et les variances de la source, l'allocation optimale par composante est approximativement $[4.27, 3.80, 3.27, 2.68, 2.10, 1.68, 1.10, 1.10]$ bits. La distorsion normalisée $D = 1/N\, E\left[\|\mathbf{x} - \hat{\mathbf{x}}\|^2\right]$ est alors proche de 0.8721, pour un rapport signal-sur-bruit de 19.64 dB.

3.6.3 Performances réelles et complexité du codage de Voronoï quasi-ellipsoïdal

Les performances optimisées et la complexité de quantification dans $g(C + \delta)$ sont présentées au tableau 3.6. La complexité de calcul est évaluée en nombre de décodages de réseau effectués. La distorsion moyenne D est définie par :

$$D = 1/N \;\; E\left[\|\mathbf{x} - gQ(\mathbf{x}/g)\|^2\right], \tag{3.33}$$

où $\mathbf{x}$ est un vecteur de source, g est le facteur d'échelle appliqué au code $C + \delta$ et Q correspond à la quantification dans $C + \delta$. La distorsion D est décomposée en deux composantes : une partie granulaire D_g et une partie de saturation (ou dépassement) D_s, avec

$$D = (1 - P_s)D_g + P_s D_s, \tag{3.34}$$

où P_s est la probabilité de saturation.

- Le gain g optimisé vaut approximativement 4.59 pour D_8 et 2.30 pour RE_8. Il est relié à la distance minimale dans le réseau, qui est l'équivalent en quantification vectorielle par réseau de points du pas de quantification scalaire uniforme. Les valeurs optimisées de g diffèrent entre D_8 et RE_8 suivant la même proportion qu'entre $d_{min}(D_8) = \sqrt{2}$ et $d_{min}(RE_8) = 2\sqrt{2}$, soit un facteur 2.

TABLEAU 3.6: Performances et complexité du codage de Voronoï ellipsoïdal.

(a) D_8

Algorithme	Performance						Complexité			Entropie
	RSB (dB)	D	P_s (%)	P_{proj} (%)	D_g	D_s	min.	moy.	max.	estimée (bits)
OPT	16.03	2.005	6.30	–	1.904	3.506	–			15.77
ORTH	15.65	2.189	6.18	2.32	1.903	6.523	2	2.84	18	17.50
RAD	15.59	2.220		2.32		7.035		2.83	18	17.49
ELLIP	15.31	2.368		5.28		9.427		2.45	18	17.46

(b) RE_8

Algorithme	Performance						Complexité			Entropie
	RSB (dB)	D	P_s (%)	P_{proj} (%)	D_g	D_s	min.	moy.	max.	estimée (bits)
OPT	16.96	1.619	8.86	–	1.519	2.652	–			15.98
ORTH	16.45	1.818	8.94	3.46	1.518	4.880	2	3.20	18	18.10
RAD	16.33	1.872		3.46		5.491		3.17	18	18.10
ELLIP	15.97	2.034		7.21		7.301		2.67	18	18.07

- La complexité réelle des algorithmes de recherche est grosso modo le double pour RE_8 par rapport à D_8, car le décodage dans RE_8 revient à décoder deux fois D_8 – en fait $2D_8$ et $2D_8 + (1, \cdots, 1)$.

  La complexité minimale s'élève toujours à 2 décodages dans Λ : pour quantifier $\mathbf{x}$ il faut effectuer au minimum un décodage dans Λ pour générer un candidat $\mathbf{y}$, puis un autre décodage pour vérifier si $\mathbf{y}$ est dans C.

- La recherche par projection et réduction (ORTH, RAD et ELLIP) avec un critère d'arrêt $\epsilon = 0.01$ de dichotomie teste au plus 9 points de $\Lambda = D_8$ ou RE_8, pour un nombre maximal de 18 décodages de Λ. Le paramètre $0 < \epsilon < 1$ permet de régler la complexité maximale de la recherche. En général $2 + \lceil -\log_2 \epsilon \rceil$ candidats sont testés, pour un nombre maximal de $4 + 2\lceil -\log_2 \epsilon \rceil$ décodages de réseau, où $\lceil . \rceil$ désigne l'arrondi à l'entier supérieur.

- La perte de performances par rapport à la recherche exhaustive due à l'emploi d'une projection varie de 0.38 à 0.72 dB pour D_8, et 0.51 à 0.99 dB pour RE_8. La probabilité de saturation P_s étant plus faible pour D_8 que pour RE_8, la performance de quantification pour D_8 est moins

sensible à la saturation que pour RE_8.

- L'estimation des entropies[2] des dictionnaires justifie la pertinence d'un codage entropique après codage de Voronoï ellipsoïdal. Le débit fixe de presque 20 bits par vecteur pourrait alors diminuer d'environ 2 bits en moyenne, mais le débit variable de la quantification compliquerait la mise en œuvre.

La valeur estimée de D est représentée en fonction du nombre de vecteurs de source à la figure 3.15 ; la recherche est effectuée sur 96,768 vecteurs avec l'algorithme OPT. L'erreur quadratique normalisée D converge assez rapidement (après $30,000$ vecteurs on ne distingue plus de fluctuations à la figure 3.15). Par contre, pour que le rapport signal signal-sur-bruit converge avec une précision de ± 0.01 dB, le nombre de vecteurs de source utilisé est tout juste suffisant. Dans le cas des algorithmes ORTH, RAD et ELLIP, les performances sont estimées de façon bien plus précise, car 9,676,800 vecteurs de source sont utilisés.

3.6.4 Comparaison avec d'autres techniques

Le compromis performances-complexité atteint par le codage de Voronoï quasi-ellipsoïdal peut être comparé à plusieurs techniques réalisables. On choisit ici pour cette comparaison :

1. La quantification scalaire non-uniforme, réalisée à partir d'une quantification scalaire uniforme dans [0,1] avec compression-expansion adaptée à une source gaussienne, comme dans [7, 9] ;
2. La quantification vectorielle non-structurée par produit cartésien.

La fonction de compression $f : \mathbb{R} \to [0,1]$ optimale *à haut débit* pour une source gaussienne scalaire x de moyenne nulle et de variance unité est donnée par [29] :

$$f(x) = \frac{1}{2}(1 + erf(x/\sqrt{6})), \tag{3.35}$$

[2] À cause de la faible taille de la base de test pour la recherche exhaustive, l'entropie mesurée pour l'algorithme OPT est sous-estimée.

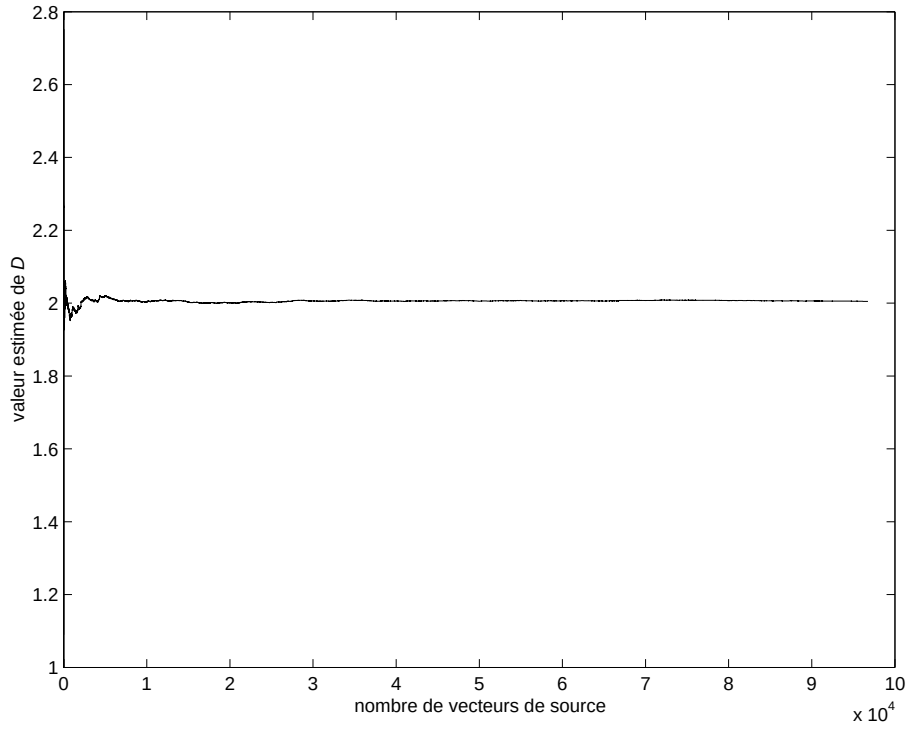


Figure 3.15: Convergence de l'erreur quadratique moyenne (normalisée) D en fonction du nombre de vecteurs de source – cas de la recherche optimale (OPT).

où *erf* est la fonction d'erreur définie par :

$$erf(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} \ dt. \tag{3.36}$$

La fonction d'expansion correspondante $f^{-1} : [0,1] \to \mathbb{R}$ est :

$$f^{-1}(y) = \sqrt{6} \ erf^{-1}(2y-1). \tag{3.37}$$

Ici la source gaussienne $\mathbf{x} = (x_1, \cdots, x_N)$ est vectorielle, de moyenne nulle et de matrice de covariance diagonale $diag(\sigma_1^2, \cdots, \sigma_N^2)$ avec $N = 8$. La quantification scalaire non-uniforme peut donc être mise en œuvre suivant le schéma de la figure 3.16. La quantification scalaire uniforme Q dans $[0,1]$ correspond à l'opération :

$$Q(x) = \frac{[xL - \frac{1}{2}] + \frac{1}{2}}{L} \tag{3.38}$$

où $L \geq 1$ est le nombre de niveaux utilisés pour quantifier x et [.] est l'arrondi à l'entier le plus proche. Le nombre de niveaux $\mathbf{L} = (L_1, \cdots, L_N)$ alloués à chaque composante scalaire de $\mathbf{x}$ peut être optimisé à partir des variances $\sigma_1^2, \cdots, \sigma_N^2$ par algorithme glouton en initialisant les composantes de $\mathbf{L}$ à 1 et avec un incrément de ± 1 ; on peut ensuite essayer de rajouter des niveaux de quantification sur chacune des composantes de $\mathbf{L}$ pour s'approcher le plus possible du budget de de bits. Pour un budget de 20 bits et avec les variances données au paragraphe 3.6.1, on trouve $\mathbf{L} = (20, 15, 10, 7, 4, 3, 2, 2)$ soit une consommation de 19.94 bits en dimension 8.

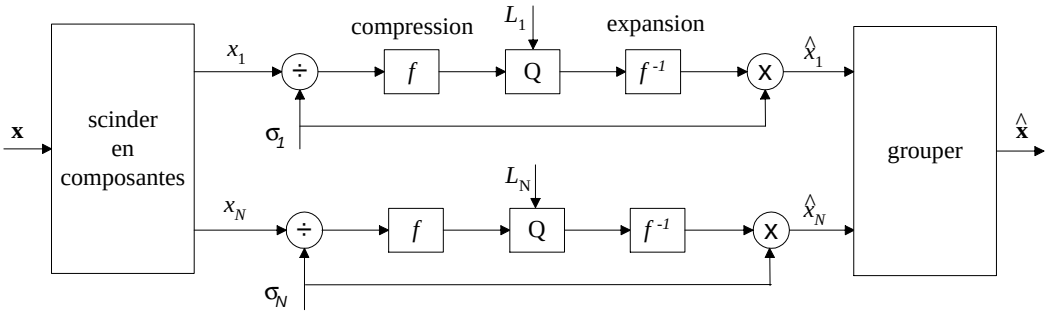


Figure 3.16: Quantification scalaire non-uniforme.

La quantification vectorielle par produit cartésien [30] est réalisée de 2 façons. Chaque vecteur de source $\mathbf{x} = (x_1, \cdots, x_8)$ est scindé en 3 sous-vecteurs de dimensions respectives 2, 2 et 4 ou bien en 2 sous-vecteurs de dimensions respectives 2 et 6. Dans le premier cas, les sous-vecteurs (x_1, x_2), (x_3, x_4) et $(x_5, \cdots, x_8)$ sont quantifiés avec respectivement 8, 6 et 6 bits. Dans l'autre cas, les sous-vecteurs (x_1, x_2) et $(x_3, \cdots, x_8)$ sont quantifiés avec 10 bits chacun. Dans les 2 cas, on utilise un total de 20 bits par vecteur, soit 2.5 bits par dimension. L'optimisation des dictionnaires suit l'algorithme LBG avec une initialisation aléatoire. La base d'apprentissage comprend 1,280,000 vecteurs de source dans le premier cas et 2,048,000 vecteurs dans l'autre cas.

Les performances testées sur 9,676,800 vecteurs de source sont comparées au tableau 3.7 avec celles présentées au paragraphe précédent. Le codage dans D_8 est légèrement moins performant que la quantification scalaire non-uniforme. Le codage dans RE_8 est par contre plus intéressant ; il permet de se rapprocher des performances de la quantification vectorielle par produit cartésien.

TABLEAU 3.7: Comparaison de performances entre le codage de Voronoï quasi-ellipsoïdal et des techniques de référence pour la source gaussienne vectorielle utilisée dans les simulations.

Technique de quantification	RSB (dB)	débit (bits/dim.)
Quantification scalaire non-uniforme	15.80	2.4929
Quantification vectorielle par produit cartésien		
dimensions 2-2-4, 8-6-6 bits	17.33	2.50
dimensions 2-6, 10-10 bits	17.18	2.50
Codage de Voronoï quasi-ellipsoïdal		
D_8, ORTH	15.65	2.4855
D_8, RAD	15.59	2.4855
RE_8, ORTH	16.45	2.4855
RE_8, RAD	16.33	2.4855

La complexité de stockage est décrite au tableau 3.8. Le codage de Voronoï quasi-ellipsoïdal et la quantification scalaire non-uniforme requiert un stockage indépendant du débit. Le stockage requis en codage de Voronoï quasi-ellispoïdal est plus faible qu'en quantification vectorielle par produit cartésien. La complexité de calcul n'est pas comparée ici, car elle implique de définir précisément une plateforme de mise en œuvre.

TABLEAU 3.8: Comparaison de complexité de stockage entre le codage de Voronoï quasi-ellipsoïdal et d'autres techniques réalisables.

Technique de quantification	Stockage
Quantification scalaire non-uniforme	8 entiers ($\mathbf{L}$), 8 flottants ($\sigma_1, \cdots, \sigma_8$), tables de calcul de erf et erf^{-1}
Quantification vectorielle par produit cartésien	
dimensions 2-2-4, 8-6-6 bits	896 flottants
dimensions 2-6, 10-10 bits	8192 flottants
Codage de Voronoï quasi-ellipsoïdal	136 entiers ($M(\Lambda)$, $M(\Lambda)^{-1}$ sous forme entière, $\mathbf{m}$), 1 flottant (g)

3.6.5 Modèle de distorsion pour le codage de Voronoï quasi-ellipsoïdal

Les résultats de simulation obtenus pour D_8 et RE_8 peuvent être prédits et interprétés à l'aide d'un modèle de distorsion. La distorsion moyenne D est décomposée comme à l'équation 3.34 à partir de la distorsion granulaire D_g, de la distorsion de saturation D_s et de la probabilité de surcharge P_s. On modélise ici séparement D_g, D_s et P_s.

En calibrant des modèles pour D_g, P_s, D_s, les performances du codage de Voronoï quasi-ellipsoïdal peuvent être estimées en fonction de g sans mettre en œuvre les algorithmes d'indexation et de recherche du plus proche voisin dans C.

Modèle de distorsion granulaire D_g

Le débit des codes utilisés en simulation étant de l'ordre 2.4855 bits par dimension, il est raisonnable pour cette erreur granulaire d'appliquer la théorie de la quantification à haut débit (*high-rate quantization theory* en anglais). Alors, en utilisant l'approximation continue, la distorsion granulaire D_g peut être modélisée comme :

$$D_g \approx Vol(V(g\Lambda))^{2/N} G(\Lambda) = g^2 (detM(\Lambda))^{2/N} G(\Lambda), \tag{3.39}$$

où $Vol(V(g\Lambda))$ est le volume de la région de Voronoï $V(\Lambda)$ de Λ (incluant le facteur d'échelle g appliqué au code), $M(\Lambda)$ est la matrice génératrice de Λ et $G(\Lambda)$ est l'erreur quadratique moyenne normalisée dans Λ (ou encore moment normalisé d'ordre 2).

On peut vérifier [14] que $G(D_8) \approx 0.075914$ et $G(RE_8) \approx 0.071682$. De plus on sait que $Vol(gV(\Lambda)) = \det(M(\Lambda))g^N$ où $M(\Lambda)$ est la matrice génératice de Λ ; avec les conditions de simulation, on a $N = 8$, $Vol(gV(D_8)) \approx 2 \times 4.59^8$ et $Vol(gV(RE_8)) \approx 256 \times 2.30^8$. La distorsion granulaire peut donc être estimée à $D_g \approx (2 \times 4.59^8)^{2/8} \times 0.075914 \approx 1.9020$ pour D_8 et $D_g \approx (256 \times 2.30^8)^{2/8} \times 0.071682 \approx 1.5168$ pour RE_8. Ces estimations sont proches des valeurs mesurées. On vérifie donc ici que $G(\Lambda)$, le moment normalisé d'ordre 2 de Λ, conditionne réellement les performances en terme de distorsion granulaire.

Modèle de probabilité de surcharge P_s et de distorsion de saturation D_s

Le débit étant de l'ordre 2.4855 bits par dimension, la probabilité de saturation P_s n'est pas négligeable. Par suite, pour que le modèle de $D = (1 - P_s)D_g + P_s D_s$ soit assez fidèle, les modèles de P_s et D_s doivent être suffisamment précis[3].

Plutôt que développer une approximation analytique de P_s et D_s comme dans [14, p. 70] et [3], on utilise ici une estimation numérique par la méthode de Monte Carlo. La frontière de surcharge (saturation) est approximée à $g(V(\Lambda') + \delta)$. Alors :

$$P_s \approx Prob\left[\mathbf{x} \notin g(V(\Lambda') + \delta)\right] = \frac{1}{\sqrt{(2\pi)^N |det(\Sigma)|}} \int_{\mathbb{R}^N \setminus \{g(V(\Lambda')+\delta)\}} e^{-\frac{1}{2}\mathbf{x}\Sigma^{-1}\mathbf{x}^t} d\mathbf{x}. \tag{3.40}$$

où $\Sigma = diag(\sigma_1^2, \cdots, \sigma_N^2)$. La distorsion de saturation peut être approximée par :

$$D_s \approx \frac{1}{N} E\left[\|\mathbf{x} - g(\mathbf{h} + \delta)\|^2 | \mathbf{x} \notin (gV(\Lambda') + \delta)\right]. \tag{3.41}$$

où $\mathbf{h}$ correspond à la projection de $\mathbf{x}/g - \delta$ sur un ellipsoïde centré sur $\mathbf{a}$. Les intégrales correspondantes peuvent être évaluées grossièrement par la méthode de Monte Carlo avec 100,000 vecteurs, à l'aide du décodage dans Λ. On trouve $P_s \approx 7.1$ % et $D_s \approx 9.40$ pour D_8 (avec $g = 4.59$) et $P_s \approx 4.8$ % et $D_s \approx 8.6$ pour RE_8 (avec $g = 2.30$).

Avec ces modèles de D_g, P_s et D_s, la distorsion D est approximée à 2.44 pour D_8 ($g = 4.59$) et 1.85 pour RE_8 ($g = 2.30$). Cette modélisation est réaliste, car ces résultats sont proches des valeurs données au tableau 3.6. Néanmoins, les modèles de P_s et D_s doivent être améliorés pour que l'optimisation du facteur d'échelle g puisse s'appuyer sur le modèle de D.

[3] Pour des débits par dimension plus élevés, la probabilité de saturation P_s tend vers 0 et la distorsion D tend à être dominée par D_g – la modélisation pour une source bien conditionnée est alors triviale.

3.7 Conclusions

Un code de Voronoï quasi-ellipsoïdal C, spécifié par un vecteur de modulo $\mathbf{m}$ et un décalage $\mathbf{a}$ dans un réseau Λ, est défini par :

$$C = \Lambda \cap \left\{ V(\Lambda') + \mathbf{a} \right\}, \tag{3.42}$$

où Λ' est un sous-réseau de Λ généré en mettant à l'échelle Λ dimension par dimension suivant les valeurs des composantes de $\mathbf{m}$. Un tel code peut être indexé en généralisant les algorithmes d'indexation de Voronoï de [1] et en incorporant au codage une étape de "modification conditionnelle". La recherche rapide du plus proche voisin dans un tel code peut être réalisée de façon sous-optimale en employant une réduction itérative et une projection adéquate. La matrice Q définie par

$$Q = M(\Lambda) diag(m_1, \cdots, m_N) M(\Lambda)^{-1}, \tag{3.43}$$

où $M(\Lambda)$ est une matrice génératice de Λ, joue un rôle majeur dans la détermination des modulos $\mathbf{m}$ admissibles et pour mettre en œuvre la modification conditionnelle. Un modulo de Voronoï $\mathbf{m}$ est admissible si $m_i \geq 2$ pour $1 \leq i \leq n$ et s'il conduit à une matrice Q entière.

Les codes de Voronoï quasi-ellipsoidaux possèdent plusieurs avantages. Ils généralisent les codes de [1]. Les algorithmes d'indexation requièrent un stockage négligeable par rapport à la quantification non structurée et indépendant du débit de codage. De plus, leur complexité algorithmique est essentiellement donnée par le décodage dans le réseau Λ et la modification conditionnelle. La mise en œuvre de l'indexation est par ailleurs relativement aisée une fois les paramètres $\mathbf{m}$ et $\mathbf{a}$ sélectionnés.

Cependant, ces codes ont également plusieurs limitations. L'étape de modification conditionnelle dépend du réseau ce qui contraste avec la généralité de [1]. En outre, pour que la troncature soit réellement quasi-ellipsoïdale, la forme de la région de Voronoï de Λ (i.e. $V(\Lambda)$) doit être aussi sphérique que possible. Par conséquent, le réseau $\mathbb{Z}$ n'est pas très approprié. Cette observation a d'ailleurs motivé le choix des réseaux A_2, D_4, $RE_8 = 2D_8^+$ et Λ_{16} qui ont de bonnes propriétés géométriques dans leur dimension respective. De plus, si l'indexation de ces codes est relativement rapide, la recherche du plus proche voisin (au sein du code) est plus problématique. En général, cette

recherche ne peut être raisonablement mise en œuvre à faible complexité qu'en utilisant une stratégie de saturation sous-optimale, (dont les performances sont à 0.5 dB de la recherche exhaustive). L'indexation de Voronoï est rapide, mais la recherche du plus proche voisin efficace est sous-optimale. Enfin, l'optimisation des performances pour la source d'un code C quasi-ellipsoïdal sous la forme $g(C+\delta)$ implique en particulier d'optimiser un facteur d'échelle g. Cette optimisation s'appuit en général sur une recherche du type descente du gradient ; un modèle de distorsion a été ébauché afin d'accélérer cette recherche.

Une application de ces codes au codage linéaire prédictif en bande élargie est présentée dans [23] où les réseaux D_{16}, $2D_{16}^{+}$ et Λ_{16} sont utilisés. Ce travail pourrait être étendu de plusieurs façons. Par exemple on pourrait :

- Considérer d'autres réseaux de points, tels que D_N^*, E_6, E_7, K_{12} et Λ_{24} – l'extension des algorithmes présentés ici est triviale une fois que l'on dispose du décodage dans Λ, d'une matrice génératrice triangulaire inférieure et des vecteurs pertinents de Λ ;
- Comparer quantitativement, en terme de performances et complexité, le codage de Voronoï quasi-ellipsoïdal et les techniques de codage algébrique ellipsoïdal de [10, 11, 12] ;
- Développer une fonction non-linéaire de compression (*companding*) pour passer d'une densité uniforme des points dans C à une densité adaptée à la quantification d'une source gaussienne, comme par exemple dans [31]. Dans [23], cette fonction de compression est approximée en dilatant C par des facteurs multiples ;
- Appliquer aux codes de Voronoï de [1] les algorithmes de recherche du plus proche voisin développés dans ce chapitre et comparer les résultats obtenus avec ceux de [3] ainsi que les estimations théoriques de [32] ;
- Généraliser les codes (ainsi que les algorithmes d'indexation associés) afin d'éliminer les contraintes d'admissibilité sur les vecteurs de modulo $\mathbf{m}$ autres que $\mathbf{m} \in (\mathbb{Z} \setminus \{0,1\})^N$;
- Trouver en fonction de Λ une définition générale du réseau entier A (spécifiant l'ensemble des modulos de Voronoï admissibles sous la forme $(\mathbb{Z} \setminus \{0,1\})^N \cap A$) ;

- Améliorer le modèle de distorsion afin d'accélérer l'optimisation du facteur d'échelle g (sans passer par une estimation coûteuse de la distorsion réelle de quantification).

Annexe 3.A : Matrices génératrices des réseaux de points utilisés

Les réseaux utilisés dans ce chapitre sont réguliers et de rang maximal. Par conséquent les matrices génératrices suivantes sont carrées. Les coefficients manquants correspondent à des zéros.

$$M(A_2) = \begin{bmatrix} 1 & 0 \\ \frac{1}{2} & \frac{\sqrt{3}}{2} \end{bmatrix} \tag{3.44}$$

Pour $N \geq 2$,

$$M(D_N) = \begin{bmatrix} 2 & & & \\ 1 & 1 & & \\ \vdots & & \ddots & \\ 1 & & & 1 \end{bmatrix} \tag{3.45}$$

À noter que $D_2 \cong \mathbb{Z}^2$ – on prend donc habituellement $N \geq 3$.

Pour N pair ≥ 4 – si N est impair, $2D_N^+$ n'est pas un réseau régulier de points [14, p. 119] –,

$$M(2D_N^+) = \begin{bmatrix} 4 & & & & \\ 2 & 2 & & & \\ \vdots & & \ddots & & \\ 2 & & & 2 & \\ 1 & 1 & \cdots & 1 & 1 \end{bmatrix} \tag{3.46}$$

À noter que $2D_4^+ \cong 2\mathbb{Z}^4$; pour utiliser un réseau performant, on prend habituellement $N \geq 8$.

La matrice $M(\Lambda_{16})$ peut être trouvée dans [14, p. 130], mais le facteur $1/\sqrt{2}$ est supprimé pour manipuler un réseau entier.

Annexe 3.B : Admissibilité d'un décalage

L'algorithme ci-après permet de vérifier de façon inductive que le décalage $\mathbf{a}$ est admissible, i.e. qu'aucun point de Λ n'est sur la frontière de $V(\Lambda') + \mathbf{a}$.

Énumération des représentants de cosets de Λ' dans Λ (vérification de l'admissibilité de a) :
Pour $\mathbf{k} \in \mathbb{Z}^N$ tel que $0 \leq k_i < m_1$ pour $1 \leq i \leq N$

- Calculer $\mathbf{x} = \mathbf{k}M(\Lambda)$
- Calculer $\mathbf{y} = \mathbf{div}(\mathbf{x} - \mathbf{a}, \mathbf{m})$
- Trouver les K proches voisins de $\mathbf{y}$ dans Λ – par exemple en utilisant un algorithme de [18]
- Si $K > 1$, arrêter (le vecteur $\mathbf{a}$ n'est pas admissible)

BIBLIOGRAPHIE

[1] J.H. Conway and N.J.A. Sloane, "A Fast Encoding Method for Lattice Codes and Quantizers," *IEEE Trans. Inform. Th.*, vol. 29, no. 6, pp. 820–824, Nov. 1983.

[2] M. Xie, *Quantification vectorielle algébrique et codage de parole en bande élargie*, Thèse de doctorat, Université de Sherbrooke, Québec, Canada, Février 1996.

[3] C. Pépin, *Quantification vectorielle et codage conjoint source-canal par les réseaux de points*, Thèse de doctorat, ENST, Paris, France, Dec. 1997.

[4] S. Ragot, M. Xie, and R. Lefebvre, "Near-Ellipsoidal Lattice Quantization by Generalized Voronoi Coding," in *Proc. IEEE Canadian Workshop on Information Theory, Vancouver, B.C., Canada*, May 2001.

[5] P. Hedelin and J. Skoglund, "Vector quantization based on gaussian mixture models," *IEEE Trans. on Speech and Audio Processing*, vol. 8, no. 4, pp. 385–401, July 2000.

[6] R.M. Gray, "Gauss mixture vector quantization," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Atlanta, GA, U.S.A.*, May 2001.

[7] A.D. Subramaniam and B.D. Rao, "PDF optimized parametric vector quantization of speech line spectral frequencies," in *Proc. IEEE Workshop on Speech Coding*, Sept. 2000, pp. 87–89.

[8] A.D. Subramaniam and B.D. Rao, "Speech LSF quantization with rate independent complexity, bit scalability and learning," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Salt Lake City, UT, U.S.A.*, 2001, vol. 2, pp. 705–708.

[9] A.D. Subramanian and B.D. Rao, "PDF Optimized Parametric Vector Quantization of Speech Line Spectral Frequencies," *IEEE Trans. on Speech and Audio Processing*, vol. 11, no. 2, pp. 130–142, Mar. 2003.

[10] T.R. Fischer, "Geometric source coding and vector quantization," *IEEE Trans. Inform. Th.*, vol. 35, no. 1, pp. 137–145, Jan. 1989.

[11] M. Barlaud, P. Solé, J.M. Moureaux, M. Antonini, , and P. Gauthier, "Elliptical codebook for lattice vector quantization," in *Proc. ICASSP*, 1993, vol. 5, pp. 590–593.

[12] J.M. Moureaux, M. Antonini, and M. Barlaud, "Counting lattice points on ellipsoids: Application to image coding," *Electon. Lett.*, vol. 31, no. 15, pp. 1224–1225, July 1995.

[13] M.J. Sabin and R.M. Gray, "Product Code Vector Quantizers for Waveform and Voice Coding," *IEEE Trans. ASSP*, vol. 32, pp. 474–488, Jun. 1984.

[14] J.H. Conway and N.J.A. Sloane, *Sphere Packings, Lattices and Groups*, Springer-Verlag, 3rd edition, 1999.

[15] G.D. Forney, "Coset Codes. I. Introduction and Geometrical Classification," *IEEE Trans. on Inf. Th.*, vol. 34, no. 5, pp. 1123–1151, Sep. 1988.

[16] G.D. Forney, "Multidimensional constellations. II. Voronoi constellations," *IEEE Trans. on Selected Areas in Communications*, vol. 7, no. 6, pp. 941–958, Aug. 1989.

[17] A. Gersho and R.M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic Publishers, 1992.

[18] E. Agrell, T. Eriksson, A. Vardy, and K. Zeger, "Closest point search in lattices," *IEEE Trans. Inform. Th.*, vol. 48, no. 8, pp. 2201–2214, Aug. 2002.

[19] T. R. Fischer, "A Pyramid Vector Quantizer," *IEEE Trans. on Information Theory*, vol. 32, no. 4, pp. 568–583, July 1986.

[20] D.G. Jeong and J.D. Gibson, "Uniform and Piecewise Uniform Lattice Vector Quantization for Memoryless Gaussian and Laplacian Sources," *IEEE Trans. Inf. Th.*, vol. 39, no. 3, pp. 786–804, May 1993.

[21] E. Viterbo and J. Boutros, "A universal lattice code decoder for fading channels," *IEEE Trans. Inf. Th.*, vol. 45, no. 5, pp. 1639–1642, Jul. 1999.

[22] G.D. Forney, "Coset Codes. II. Binary Lattices and related codes," *IEEE Trans. on Inf. Th.*, vol. 34, no. 5, pp. 1152–1187, Sep. 1988.

[23] S. Ragot, H. Lahdili, and R. Lefebvre, "Wideband LSF quantization by generalized Voronoi codes," in *Proceedings of Eurospeech, Aalborg, Denmark*, Sep. 2001, pp. 2319–2322.

[24] C. Lamblin and J.-P. Adoul, "Algorithme de quantification vectorielle sphérique à partir du réseau de Gosset d'ordre 8," *Ann. Télécommun.*, vol. 43, no. 3–4, pp. 172–186, 1988.

[25] E. Agrell, "On Voronoi-relevant vectors," private communication, 2002.

[26] Y. Huang and P.M. Schultheiss, "Block quantization of correlated gaussian random variables," *IEEE Trans. Commun. Syst.*, vol. C-10, pp. 289–296, Sep. 1963.

[27] D.J. Sakrison, "A geometric treatment of the source encoding of a Gaussian random variable," *IEEE Trans. Inf. Th.*, vol. IT-14, pp. 481–486, May 1968.

[28] W.H. Press, *Numerical recipes in C : the art of scientific computing*, Cambridge University Press, 1992.

[29] J.A. Bucklew and N.C. Gallagher, "A note on the computation of optimal minimum mean-square error quantizers," *IEEE Trans. Commun.*, vol. 30, no. 1, pp. 298–301, 1982.

[30] K. Paliwal and B.S. Atal, "Efficient Vector Quantization of LPC Parameters at 24Bits/Frame," *IEEE Trans. on Speech and Audio Processing*, vol. 1, Jan. 1993.

[31] T.Z. Shabestary and P. Hedelin, "Spectral quantization by companding," in *Proc. ICASSP*, 2002, vol. 1, pp. 641–644.

[32] M.V. Eyuboglu and G.D. Forney, "Lattice and Trellis Quantization with Lattice- and Trellis-Bounded Codebooks – High-Rate Theory for Memoryless Sources," *IEEE Trans. on Inf. Th.*, vol. 39, no. 1, pp. 46–59, Jan. 1993.

[33] U. Flincke and N. Pohst, "Improved Methods for Calculating Vectors of Short Length in a Lattice, Including a Complexity Analysis," *Mathematics of Computation*, vol. 44, no. 170, pp. 463–471, Apr. 1985.

[34] A. Vardy and Y. Be'ery, "Maximum likelihood decoding of the Leech lattice," *IEEE Trans. Inf. Th.*, vol. 39, no. 4, pp. 1435–1444, Jul. 1993.

[35] J.H. Conway and N.J.A. Sloane, "Fast quantizing and decoding algorithms for lattice quantizers and codes," *IEEE Trans. Inform. Th.*, vol. 28, pp. 227–232, Mar. 1982.

[36] J.-P. Adoul and M. Barth, "Nearest neighbor algorithm for spherical codes from the Leech lattice," *IEEE Transactions on Information Theory*, vol. 34, no. 5, pp. 1188–1202, Sep. 1988.

[37] M. Xie and J.-P. Adoul, "Embedded algebraic vector quantizer (EAVQ) with application to wideband speech coding," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Atlanta, GA, U.S.A.*, 7-10 May 1996, vol. 1, pp. 240–243.

[38] T.D. Lookabaugh and R.M. Gray, "High-resolution quantization theory and the vector quantizer advantage," *IEEE Transactions on Information Theory*, vol. 35, no. 5, pp. 1020–1033, Sep. 1989.

[39] M. Barlaud, P. Solé, T. Gaidon, M. Antonini, and P. Mathieu, "Pyramidal Lattice Vector Quantization for Multiscale Image Coding," *IEEE Trans. on Image Processing*, vol. 3, no. 4, pp. 367–381, July 1994.

Chapitre 4

EXTENSION DE VORONOÏ

Ce chapitre est consacré à une nouvelle technique de quantification multi-débit, développée à l'origine comme une extension de la quantification vectorielle algébrique imbriquée introduite dans [1]. La quantification multi-débit présentée ici repose sur une méthode, appelée extension de Voronoï, due à Adoul ; celle-ci a été mise en œuve sous forme algorithmique dès 1996, sans toutefois avoir été publiée. Ce chapitre apporte plusieurs contributions à cette technique. Tout d'abord, le concept d'extension de Voronoï y est défini précisément à partir de la la théorie des codes à cosets de Forney [2, 3, 4]. Cette définition rigoureuse n'a jamais été clarifiée ni expliquée auparavant. On décrit ensuite comment l'extension de Voronoï peut être utilisée en quantification en décrivant un algorithme général de quantification multi-débit. Cette description inédite est complétée par une étude de performances qui valide l'intérêt de la quantification multi-débit par extension de Voronoï.

L'organisation du chapitre est la suivante. Le problème de la quantification multi-débit pour le codage TCX est posé à la section 4.1 ; la technique de quantification de [1] y est en particulier analysée. L'extension de Voronoï est définie précisément dans la section 4.2. Les algorithmes de codage et de décodage permettant de mettre en œuvre la quantification algébrique multi-débit par extension de Voronoï sont explicités dans la section 4.3. La quantification multi-débit par extension de Voronoï est évaluée quantitativement pour les réseaux A_2 et RE_8 dans le cas d'une

source gaussienne sans mémoire dans la section 4.4. Un système complet de quantification en dimension 8 (à partir du réseau RE_8), étendant celui de [1] et appliqué en codage ACELP/TCX multi-mode [5] est décrit ensuite dans la section 4.5. L'extension de Voronoï est généralisée à la section 4.6, avant de conclure dans la section 4.7.

4.1 Problématique : quantification multi-débit pour le codage TCX des signaux de parole et de musique

Le développement de l'extension de Voronoï a été motivé par le problème du codage par blocs de la cible TCX à débit fixe. En codage TCX [6], la cible est définie comme le signal temporel pondéré par le filtre perceptuel $\hat{W}(z)$, duquel on retranche la mémoire du filtre $1/\hat{A}(z/\gamma)$ de la trame passée. Cette mémoire, appelée *ringing* en anglais, est la réponse du filtre $1/\hat{A}(z/\gamma)$ à une entrée nulle. Le filtre $\hat{W}(z)$ est généralement défini comme [6]

$$\hat{W}(z) = \frac{\hat{A}(z)}{\hat{A}(z/\gamma)}, \tag{4.1}$$

où $\hat{A}(z) = 1+\hat{a}_1 z^{-1}+\cdots+\hat{a}_p z^{-p}$ est un filtre de prédiction linéaire d'ordre p (quantifié) et $0 \leq \gamma \leq 1$ un facteur appelé facteur perceptuel (généralement de l'ordre de 0.75). Dans le contexte du codage ACELP/TCX multi-mode [5], la prédiction de pitch n'est pas utilisée. Une vue simplifiée du codeur est illustrée à la figure 4.1. La cible TCX est décomposée par transformation linéaire T pour former un bloc (vecteur) de coefficients transformés, avant d'être codée.

4.1.1 Codage de la cible TCX

La quantification de la cible TCX dépend de la transformation T. On se restreint ici au cas où T est une transformation de Fourier discrète comme dans [1, 7, 8, 9]. D'autres transformations sont utilisées dans [10, 11, 12, 13]. Le problème du codage par transformée de Fourier de la cible TCX a été abordé de plusieurs façons :

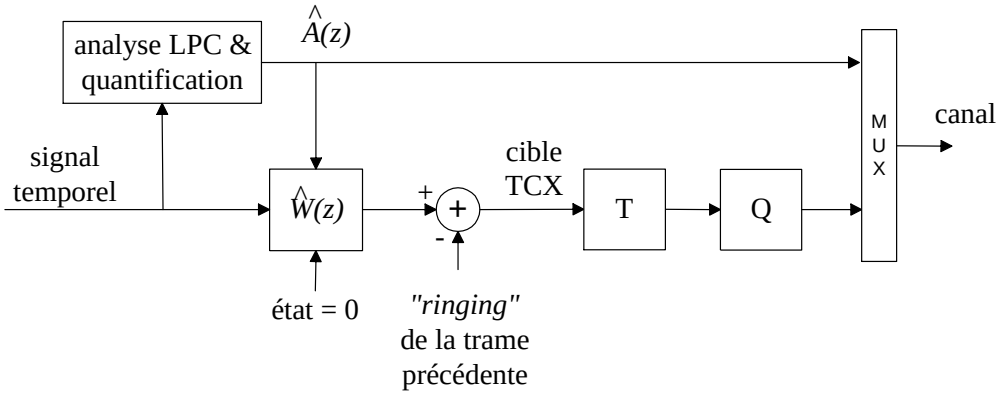


Figure 4.1: Modèle simplifié du codage TCX sans prédiction de pitch.

- Une quantification polaire [7, 8, 9, 6] de type amplitude-phase. Cette approche permet d'exploiter la corrélation intertrame sur les amplitudes, la phase (distribuée uniformément) étant codée en fonction de l'amplitude ;

- Une quantification cartésienne de type gain-forme où le spectre, normalisé par un gain global, est représenté par quantification vectorielle algébrique par produit cartésien [1]. Des dictionnaires algébriques de différents débits sont alors nécessaires pour coder chaque sous-vecteur ; le codage des sous-vecteurs est réalisé dans [1] par quantification vectorielle algébrique imbriquée (*embedded algebraic vector quantization*).

Cette seconde approche offre plusieurs avantages : faible complexité, minimisation directe de l'erreur quadratique et découplage de la quantification et de la mise en forme du bruit de codage. L'erreur de quantification est approximativement de niveau uniforme (plate) sur l'ensemble du spectre car le pas de quantification est identique pour chaque sous-vecteur ; la forme du bruit de quantification suit la réponse du filtre perceptuel inverse $\hat{W}(z)^{-1}$.

4.1.2 Limitations de la quantification vectorielle algébrique imbriquée

La technique de quantification multi-débit de [1] est une technique de quantification vectorielle algébrique en dimension 8, développée dans le contexte du codage TCX avec prédiction de pitch de parole en bande élargie. Elle utilise 6 dictionnaires de quantification, $Q_0, Q_1, Q_2, \cdots, Q_5$, où Q_0 ne contient que le mot nul, et les 5 autres dictionnaires sont imbriqués (c'est-à-dire $Q_1 \subset Q_2 \subset \cdots \subset Q_5$). Pour un n fixé, Q_n comprend 2^{4n} mots de code pour un débit de $n/2$ bit par dimension. La technique introduite dans [1] s'est avérée efficace pour le codage de la parole en bande élargie à 16 kbit/s et une longueur de trame de 20 ms. Cependant elle possède plusieurs inconvénients :

1. Limitation d'allocation des bits : Elle ne permet pas d'allouer plus de 20 bits par vecteur en dimension 8. Or, en codage par transformée (en particulier à faible recouvrement), les vecteurs de plus grande énergie – les plus enclins à causer une surcharge – doivent normalement être quantifiés avec une distorsion faible pour maximiser la qualité et limiter les effets de trame. Pour un codage TCX sans prédiction de pitch avec des trames de plus de 20 ms et des signaux de musique, cette contrainte réduit (a priori) fortement la qualité de codage car les sous-vecteurs de forte énergie sont alors mal codés.

   De plus, la limitation de l'allocation des bits à 20 bits est spécifique au contexte de [1] de codage à 16 kbit/s et n'est a priori pas adaptée au codage à débit élevé (par exemple à 24 ou 32 kbit/s).

2. Complexité de calcul : Parce que l'allocation des bits est limitée, une procédure de saturation doit être utilisée en cas de surcharge dans le dictionnaire Q_5 de 20 bits. Dans [1], la saturation consiste à réduire le vecteur à quantifier homothétiquement jusqu'à ce qu'il n'y ait plus de surcharge. D'autres techniques de saturation sont possibles. Mais dans tous les cas, la saturation est complexe en terme de quantité de calculs. De plus, elle dégrade significativement les performances (donc la qualité) en cas de grand dépassement.

3. Complexité de stockage : Les dictionnaires Q_2, Q_3, Q_4 et Q_5 de 8, 12, 16 et 20 bits respectivement sont spécifiés par 3, 8, 23 et 73 leaders absolus. Puisque l'espace de stockage et

la complexité de calcul de la recherche sont liés au nombre de leaders absolus, la complexité des dictionnaires explose avec le débit du dictionnaire. De plus l'utilisation d'un dictionnaire de 20 bits oblige à stocker certains paramètres d'indexation sur 32 bits ce qui augmente la complexité de stockage.

L'extension de Voronoï permet de régler tous ces problèmes.

La quantification vectorielle algébrique imbriquée comporte par ailleurs une autre limitation pour le codage TCX audio : elle a été developpée en supposant que les sous-vecteurs (de dimension identique) de la cible TCX transformée ont une distribution gaussienne i.i.d. Cette hypothèse est valable pour un modèle TCX avec prédiction de pitch et une taille de trame de 20 ms (résultant en un fort étalement spectral). En codage TCX audio, la prédiction de pitch n'est pas employée, la taille de trames varie généralement entre 20 et 80 ms.

4.2 Extension de Voronoï : définition et propriétés

4.2.1 Préliminaires : codage de Voronoï et partitionnement de réseaux de points

Les codes de Voronoï ont été introduits dans [14] comme des constellations de points optimisées pour une indexation efficace. Pour un réseau de points Λ en dimension N et un entier $m \geq 2$, un code de Voronoï $V(\Lambda, m\Lambda)$ de taille m^N est généré en tronquant Λ par la région de Voronoï $V(\Lambda)$ de Λ associée à l'origine, mise à l'échelle par m et décalée de $\mathbf{a} \in \mathbb{R}^N$:

$$V(\Lambda, m\Lambda) = \Lambda \cap (mV(\Lambda) + \mathbf{a}) = \Lambda \cap (V(m\Lambda) + \mathbf{a}) \,. \tag{4.2}$$

Le décalage $\mathbf{a}$ sert à s'assurer qu'aucun point de Λ ne tombe sur la surface de la région de troncature. Des algorithmes d'indexation pour de tels codes sont présentés dans [14].

La définition des codes de Voronoï a été généralisée dans [4] au cas de la troncature (mise en forme) par une région de Voronoï d'un sous-réseau Λ' de Λ. La taille du code résultant correspond

au nombre de cosets de Λ' dans Λ. Si $\Lambda' = 2^r\Lambda$, où r est un entier ≥ 1, il y a 2^{Nr} cosets, c'est-à-dire qu'on peut générer Λ par 2^{Nr} translations distinctes de Λ'.

Tel qu'indiqué dans [4, p.941], un code de Voronoi $V(\Lambda, \Lambda')$ permet de générer tous les cosets de la partition Λ/Λ' à partir de Λ'. Cette propríété peut être résumée par la formule de code :

$$\Lambda = \Lambda' + V(\Lambda, \Lambda') = \bigcup_{\mathbf{v} \in V(\Lambda, \Lambda')} \left(\Lambda' + \mathbf{v}\right). \tag{4.3}$$

Cette propriété fondamentale est exploitée ici pour étendre les constellations de points issues de Λ. Elle s'explique simplement à partir de l'indexation de Voronoï. Les m^N points d'un code de Voronoï sont générés sous la forme :

$$\mathbf{v} = \mathbf{k}M(\Lambda) - m\mathbf{z}, \tag{4.4}$$

où $\mathbf{k} = (k_1, \cdots, k_N)$ est un vecteur entier tel que $0 \leq k_i < m$ pour $i = 1, \cdots, N$ et $\mathbf{z}$ est le plus proche voisin de $\frac{1}{m}(\mathbf{k}M(\Lambda) - a)$ dans Λ. L'indice $\mathbf{k}$ définit dans quel coset de $m\Lambda$ dans Λ se trouve le mot de code $\mathbf{v}$.

4.2.2 Définition théorique : extension à partir d'une partition $\Lambda/2^r\Lambda$

Pour un réseau Λ en dimension N et un code C issu de Λ de débit par dimension R_C (comprenant 2^{NR_C} points), l'extension de Voronoï $C^{(r)}$ de C d'ordre r est définie par :

$$C^{(r)} = 2^rC + V(\Lambda, 2^r\Lambda) = \bigcup_{\substack{\mathbf{c} \in C \\ \mathbf{v} \in V(\Lambda, 2^r\Lambda)}} \left(2^r\mathbf{c} + \mathbf{v}\right), \tag{4.5}$$

où r est un entier ≥ 1. Le décalage $\mathbf{a}$ définissant $V(\Lambda, 2^r\Lambda)$ est identique pour chaque r. Par convention, on prend $C^{(0)} = C$.

Par définition, le code étendu $C^{(r)}$ requiert NR_C bits pour C plus Nr bits pour $V(\Lambda, 2^r\Lambda)$. Par suite, le code résultant $C^{(r)}$ est de débit par dimension $R_C + r$. La granularité en débit de cette forme d'extension est donc de 1 bit par dimension – le débit du code est étendu de 1 bit par dimension en passant de l'extension d'ordre r à $r + 1$.

L'extension consiste donc à mettre à l'échelle le code C par des puissances successives de 2 (2, 4, 8, etc.) et à insérer un code de Voronoï autour de chaque point du dictionnaire de base C dilaté. Pour cette raison, l'extension est appelée ici "extension de Voronoï". On peut aussi voir l'extension de Voronoï comme technique de codage multi-étage (*multistage quantization*), où le code $2^r C$ est le dictionnaire du premier étage et le code $V(\Lambda, 2^r\Lambda)$ est le dictionnaire du second étage.

Exemple : Ce mécanisme est illustré à la figure 4.2 à l'aide du réseau A_2. Le réseau infini A_2 peut être défini par :

$$A_2 = \mathbb{Z}\mathbf{v}_1 + \mathbb{Z}\mathbf{v}_2 \tag{4.6}$$

où $\mathbf{v}_1 = (1, 0)$ et $\mathbf{v}_2 = (1/2, \sqrt{3}/2)$. Une constellation de points peut être obtenue à partir de A_2 en tronquant adéquatement le réseau pour n'en retenir qu'un sous-ensemble fini de points. On choisit ici une troncature sphérique telle qu'à la figure 4.2 (a). Seuls les points de A_2 à l'intérieur de la région de troncature (une sphère centrée de rayon $2\sqrt{2}$) sont retenus pour former le code C :

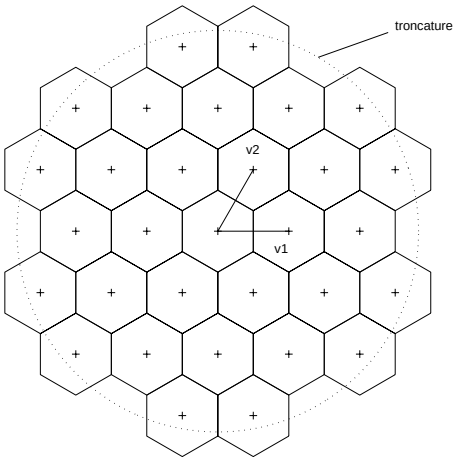
$$C = \left\{ \mathbf{y} \in A_2 \;\mid\; \|\mathbf{y}\| \leq 2\sqrt{2} \right\}. \tag{4.7}$$

Le dictionnaire C comprend les couches sphériques 0 à 7 de A_2 et contient 31 points. Le code étendu $C^{(1)} = 2C + V(A_2, 2A_2)$ est montré à la figure 4.2 (b), avec $\mathbf{a} = (0.1, 0)$. Les croix (+) identifient le code C mis à l'échelle par un facteur 2. Chaque mot $\mathbf{y}$ de $C^{(1)}$ peut être décomposé sous la forme $\mathbf{y} = 2\mathbf{c} + \mathbf{v}$ où $\mathbf{c} \in C$ et $\mathbf{v} \in V(A_2, 2A_2)$. Le code $C^{(1)}$ comprend 4 fois plus de mots de code que C.

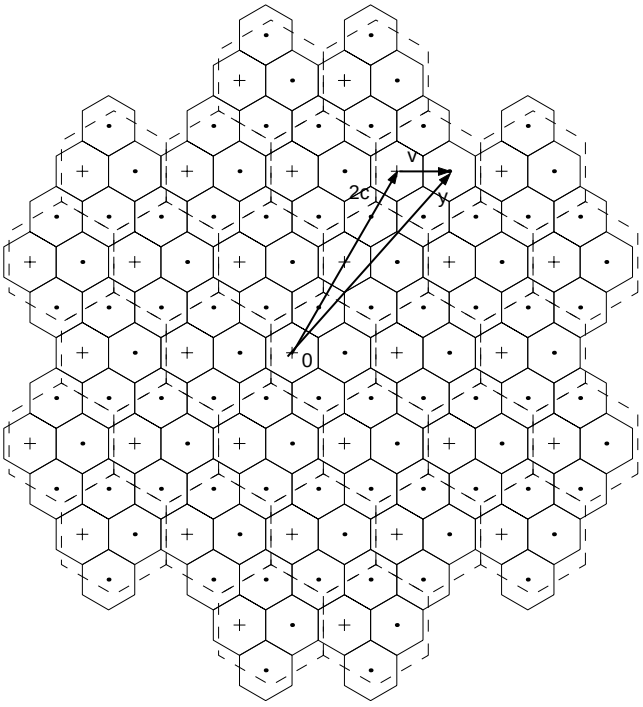
Les codes C, $C^{(1)}$ et $C^{(2)}$ sont illustrés à la figure 4.3. Les axes indiquent le repère orthonormé du plan (les échelles sont identiques sur chacune des figures). Cette figure permet de vérifier que l'extension préserve la granularité (ou résolution) ainsi que la forme approximative – de la région granulaire – de la constellation originale.

4.2.3 Propriétés

L'extension de Voronoï possède plusieurs propriétés intéressantes enumérées:

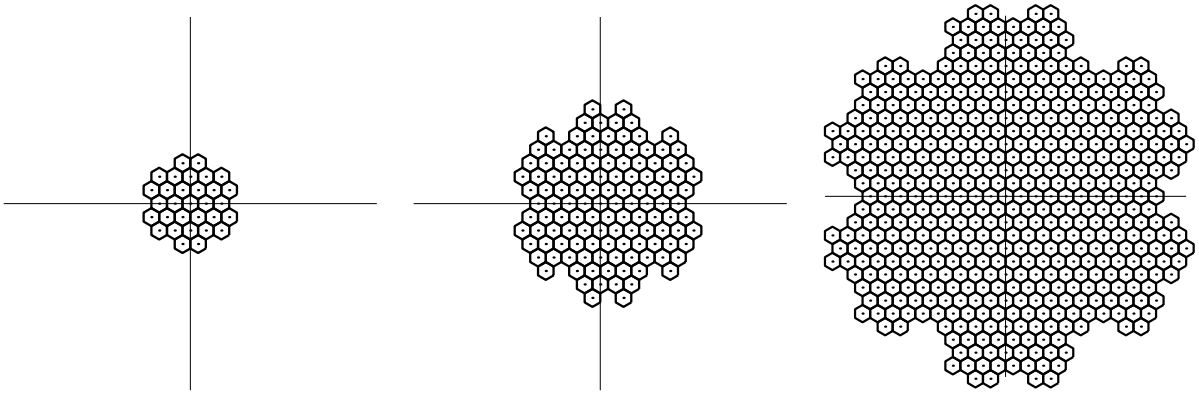


(a) code C obtenu par troncature sphérique de A_2



(b) code $C^{(1)} = 2C + V(A_2, 2A_2)$ avec $\mathbf{a} = (0.1, 0)$

Figure 4.2: Exemple d'extension de Voronoï pour le réseau A_2.



(a) constellation C (b) constellation C après extension (+2 bits) (c) constellation C après extension (+4 bits)

Figure 4.3: Exemple d'extension de Voronoï pour le réseau A_2.

1. Elle est valable pour n'importe quel réseau régulier de points Λ (y compris le réseau RE_8 utilisé dans [1]).

2. Une constellation étendue $C^{(r)}$ comprend 2^{Nr} fois plus de points que la constellation de base C. Ainsi, l'extension permet de construire des codes de débit plus élevé que C.

3. Si une constellation C de Λ ($C \subset \Lambda$) est étendue à l'aide d'un code $V(\Lambda, 2^r\Lambda)$, $C^{(r)}$ est une constellation de Λ ($C^{(r)} \subset \Lambda$).

4. Conservation de la granularité et de la forme : Si le code C est généré en tronquant Λ par une région $\mathcal{R}$, i.e. $C = \Lambda \cap \mathcal{R}$, le code $C^{(r)}$ peut être approximativement vu comme généré par une troncature de Λ par $2^r\mathcal{R} + \mathbf{a}$. Les formes des constellations C et $C^{(r)}$ (c'est-à-dire les frontières des zones granulaires de ces codes) sont donc similaires au facteur d'échelle 2^r près.

5. Si le code C est de moyenne nulle :

   - Le centroïde des codes $C^{(r)}$ est un point proche du décalage $\mathbf{a}$. En choisissant $\mathbf{a}$ de sorte que $\|\mathbf{a}\| \approx 0$, les codes $C^{(r)}$ sont alors de moyenne quasi-nulle.
   - L'énergie moyenne de $C^{(r)}$ est la somme des énergies moyennes de 2^rC et du code de

Voronoï $V(\Lambda, 2^r\Lambda)$, soit:

$$\underbrace{\left[\frac{1}{2^{N(R_C+r)}}\sum_{\mathbf{y}\in C^{(r)}}\|\mathbf{y}\|^2\right]}_{\text{énergie moyenne de } C^{(r)}} = 2^{2r}\ \underbrace{\left[\frac{1}{2^{NR_C}}\sum_{\mathbf{c}\in C}\|\mathbf{c}\|^2\right]}_{\text{énergie moyenne de } C} + \underbrace{\left[\frac{1}{2^{Nr}}\sum_{\mathbf{v}\in V(\Lambda,2^r\Lambda)}\|\mathbf{v}\|^2\right]}_{\text{énergie moyenne de } V(\Lambda,2^r\Lambda)}. \tag{4.8}$$

En supposant que $V(\Lambda, 2^r\Lambda)$ est de moyenne nulle, l'énergie moyenne du $V(\Lambda, 2^r\Lambda)$ est proche de $NG(2^r\Lambda)$, où $G(2^r\Lambda)$ est le moment normalisé d'ordre 2 de $2^r\Lambda$. Cette énergie peut se développer sous la forme $NG(2^r\Lambda) = 2^{2r}N(\det M(\Lambda))^{2/N}G(\Lambda)$, où $G(\Lambda)$ est le moment normalisé d'ordre 2 de Λ. En fait, le code $V(\Lambda, 2^r\Lambda)$ est plutôt centré en un point proche de $\mathbf{a}$ – le décalage définissant $V(\Lambda, 2^r\Lambda)$. L'énergie moyenne de $C^{(r)}$ peut être approximée à :

$$\frac{1}{2^{N(R_C+r)}}\sum_{\mathbf{y}\in C^{(r)}}\|\mathbf{y}\|^2 \approx 2^{2r}\left(\left[\frac{1}{2^{NR_C}}\sum_{\mathbf{c}\in C}\|\mathbf{c}\|^2\right] + N(\det M(\Lambda))^{2/N}G(\Lambda)\right) + \|\mathbf{a}\|^2. \tag{4.9}$$

Pour chaque bit supplémentaire (par dimension), l'énergie du code est donc approximativement multipliée par 4 – conformément à la théorie de la distorsion [15].

6. L'indice d'un point dans $C^{(r)}$ est implicitement séparé en deux composantes : un indice de taille fixe dans C (de NR_C bits par vecteur) et un indice de Voronoï (de Nr bits par vecteur). Cette propriété permet d'améliorer la robustesse du code contre les erreurs binaires. De ce point de vue, un code étendu par extension de Voronoï est comparable à un code à descriptions multiples asymétrique [16] ou à structure conjuguée [17].

7. Sous certaines conditions, l'extension de Voronoï produit une série de codes imbriqués, avec $C \subset C^{(1)} \subset C^{(2)} \subset \ldots$ et asymptotiquement $C^{(+\infty)} = \Lambda$. Cette propriété est par exemple vérifiée si C est un code générée par troncature sphérique de Λ centrée sur l'origine – avec une sphère de troncature de rayon suffisant – et si le décalage $\mathbf{a}$ intervenant dans la définition de $V(\Lambda, 2^r\Lambda)$ est proche de l'origine.

4.3 Quantification multi-débit par extension de Voronoï

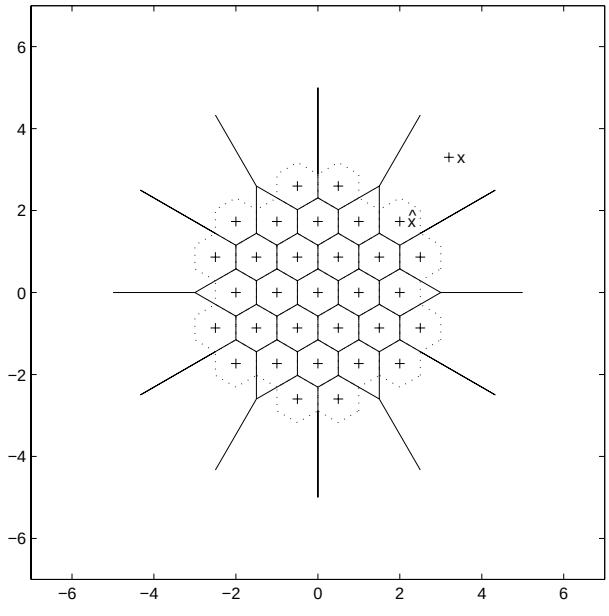
4.3.1 Exemple

La figure 4.4 montre un exemple de quantification basée sur l'extension de Voronoï. Le code C est le même que celui montré à la figure 4.2 (a). Il comprend 31 points ; il est indexé avec 5 bits.

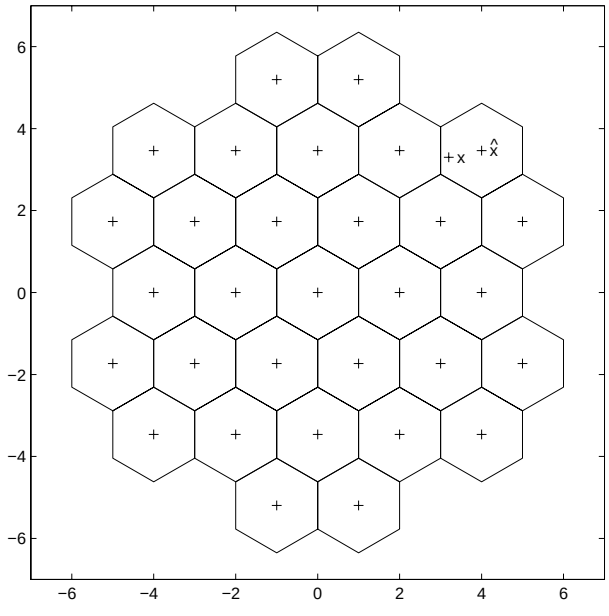
A la figure 4.4 (a), un vecteur de source $\mathbf{x}$ est représenté. De toute évidence, le plus proche voisin $\mathbf{y}$ de $\mathbf{x}$ dans A_2 n'est pas un mot de code de C. Autrement dit, $\mathbf{x}$ cause une surcharge dans C. Pour éviter un dépassement, une solution consisterait à "dilater" la constellation C en la multipliant par exemple par 2 – comme indiqué à la figure 4.4 (b). Le plus proche voisin de $\mathbf{x}$ dans $2A_2$ serait alors un mot du code $2C$, mais les régions de Voronoï seraient de plus grandes tailles. Cette solution augmente donc la distorsion granulaire.

Pour conserver la même taille de région de Voronoï et donc maintenir la même granularité que celle de C tout en étendant la frontière de surcharge du code, on applique une extension de Voronoï d'ordre 1 à C. Autrement dit, le code C est mis à l'échelle par un facteur 2 et un code de Voronoï est inséré autour de chaque mot de code de $2C$, tel qu'indiqué à la figure 4.4 (c). Le code de Voronoï est $V(A_2, 2A_2) = A_2 \cap \{2V(A_2) + \mathbf{a}\}$, avec $\mathbf{a} = (0.1, 0)$. Il comprend ici 4 points et requiert donc 2 bits additionnels. Le dictionnaire étendu $C^{(1)}$ permet d'éliminer la surcharge tout en conservant la même granularité. A la figure 4.4 (c), il apparaît que le plus proche voisin $\mathbf{y}$ de $\mathbf{x}$ dans A_2 appartient à $C^{(1)}$. Cependant pour décrire $\mathbf{y}$ dans $C^{(1)}$, on devra utiliser 5+2=7 bits au lieu des 5 bits du dictionnaire de base C. De plus une information doit être transmise pour indiquer que $C^{(1)}$ est utilisé au lieu de C.

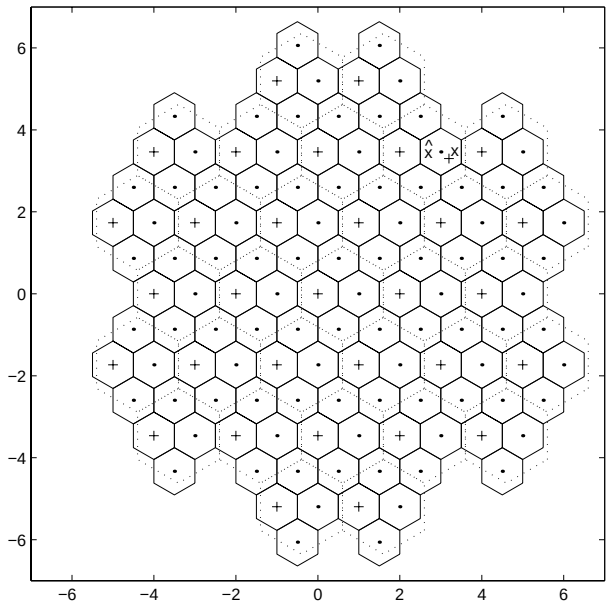
Comme le montre la figure 4.4 (d), le vecteur $\mathbf{y}$ peut être décomposé comme $\mathbf{y} = m\mathbf{c} + \mathbf{v}$, où m est le facteur d'échelle appliqué (ici $m = 2$), $\mathbf{c}$ est un mot du code dans C et $\mathbf{v}$ est un mot de code de Voronoï. Cet exemple est généralisé ici pour concevoir une quantification multi-débit sans saturation.



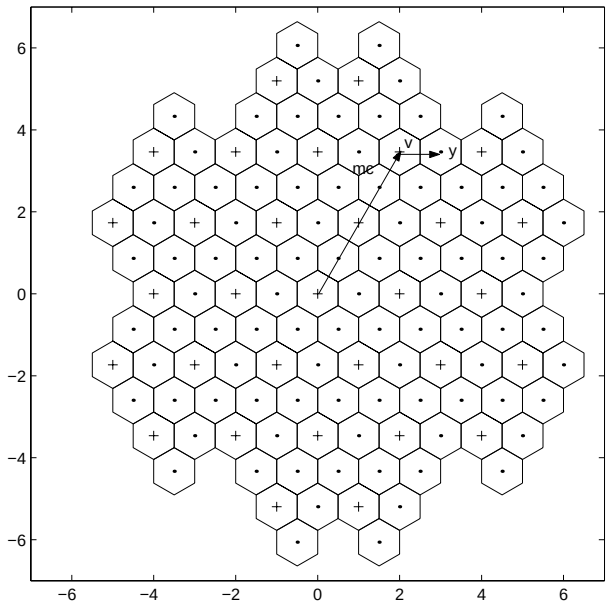
(a) quantification de $\mathbf{x}$ dans C



(a) quantification de $\mathbf{x}$ dans $2C$



(c) quantification de $\mathbf{x}$ dans $C^(1)$



(d) décomposition de $\hat{\mathbf{x}} = \mathbf{y} = m\mathbf{c} + \mathbf{v}$

Figure 4.4: Exemple de quantification.

4.3.2 Algorithme de quantification (codage et décodage)

Le codage décrit dans l'exemple précédent est formalisé à la figure 4.5. Un vecteur de source $\mathbf{x}$ est représenté dans un réseau de points Λ. Le plus proche voisin $\mathbf{y}$ de $\mathbf{x}$ est trouvé dans Λ. Le problème du codage de $\mathbf{x}$ est réduit à indexer $\mathbf{y}$ par i et n pour reconstruire exactement $\mathbf{y}$. L'indice i identifie un mot de code dans un dictionnaire de numéro n. Pour un code C de débit fixe issu de Λ, on peut prendre $n = r$ si $y \in C^{(r)}$, où $r \geq 0$. Le numéro n peut donc être interprété comme un numéro de dictionnaire indiquant l'ordre d'extension de C.

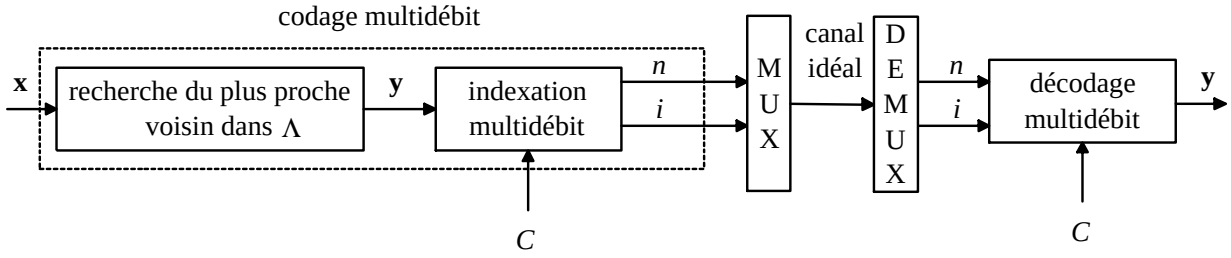


Figure 4.5: Principe de la quantification multi-débit par extension de Voronoï.

Algorithme de codage

Les étapes du codage de $\mathbf{x}$ dans $C^{(r)}$, où $r \geq 0$, sont détaillées ci-dessous :

1) Trouver le plus proche voisin $\mathbf{y}$ de $\mathbf{x}$ dans Λ. (En fait $\mathbf{y}$ est le point qui sera reconstruit par le décodeur. L'objectif du codage est donc d'indexer adéquatement $\mathbf{y}$).

2) Si $\mathbf{y} \in C$, calculer l'indice i de $\mathbf{y}$ en tant que mot de code dans C. Fixer le numéro de dictionnaire $n = 0$. Fin. (Le codage de $\mathbf{x}$ requiert alors NR_C bits pour i, plus l'indication que $\mathbf{y}$ est dans le dictionnaire de base).

   Autrement, si $\mathbf{y} \notin C$, $\mathbf{y}$ occasionne une surcharge. On utilise plus de bits pour décrire $\mathbf{y}$ et l'extension de Voronoï est appliquée :

(i) Initialisation : fixer $r = 1$ et $m = 2^r = 2$.

(ii) Chercher le plus proche voisin $\mathbf{c}$ de $(\mathbf{y} - \mathbf{a})/m$ dans Λ.

(iii) Si $\mathbf{c} \notin C$, incrémenter r de 1 et multiplier m par 2 ; aller à l'étape (ii).

(iv) Si $\mathbf{c} \in C$, l'ordre d'extension r est suffisant pour quantifier $\mathbf{x}$ en $\mathbf{y}$ sans saturation dans $C^{(r)}$ sous la forme $\mathbf{y} = m\mathbf{c} + \mathbf{v}$ où $\mathbf{c} \in C$ et $\mathbf{v} \in V(\lambda, 2^r\Lambda)$. Fixer le numéro de dictionnaire $n = r$. Calculer l'indice j de $\mathbf{c}$ dans C. Calculer l'indice de Voronoï $\mathbf{k}$ de $\mathbf{v}$ par modulo :

- calculer $\mathbf{l} = \mathbf{y}M(\Lambda)^{-1}$;
- calculer $\mathbf{k} = (l_1 \ mod \ m, \cdots, l_N \ mod \ m)$.

Le calcul de $\mathbf{k}$ est tiré de [14] ; il est égalemetn décrit au chapitre 2. Multiplexer j et $\mathbf{k}$ pour former l'indice i. On rappelle que j et $\mathbf{k}$ sont représentés sur respectivement NR_C et Nr bits.

Le codage de $\mathbf{x}$ genère un numéro de dictionnaire n et un indice i représenté sur $N(R_C + r)$ bits, avec $n = r$. Par convention, si l'extension de Voronoï est utilisée, $n > 0$, sinon $n = 0$.

L'indice n est entier et doit être représenté adéquatement sous forme binaire. Un exemple est donné à la section 4.4, où n est représenté par codage unaire. En général, n et i doivent être codés sans perte afin de réduire le débit moyen de codage.

Algorithme de décodage

Le numéro de dictionnaire n est lu à partir du canal. Il est en fait obtenu en décodant la représentation binaire de n utilisée au codage. Puisque n indique implicitement le nombre de bits alloués à l'indice i, il est essentiel d'en connaître la valeur pour démultiplexer correctement i. Ensuite :

- Si $n = 0$, l'extension de Voronoï n'est pas utilisée. Dans ce cas, l'indice i est décodé pour former un mot de code $\mathbf{c}$ du dictionnaire de base C. Le vecteur reconstruit est simplement $\mathbf{y} = \mathbf{c}$.

- Si $n > 0$, l'extension de Voronoï est utilisée. L'ordre d'extension est fixé à $r = n$ et le modulo de Voronoï à $m = 2^r$. Étant donné n, les indices j et $\mathbf{k}$ sont démultiplexés du canal. L'indice j est décodé pour former le mot de code $\mathbf{c}$ dans C, tandis que $\mathbf{k}$ est décodé en $\mathbf{v}$ dans le code de Voronoï $V(\Lambda, 2^r\Lambda)$. Le vecteur reconstruit est alors donné par $\mathbf{y} = m\mathbf{c} + \mathbf{v}$.

4.3.3 Remarques importantes

1. L'extension de Voronoï est utilisée uniquement quand le plus proche voisin $\mathbf{y}$ de $\mathbf{x}$ n'est pas un mot de code dans le dictionnaire de base C. Pourvu que le budget de bits soit suffisant, $\mathbf{y}$ est indexé dans $C^{(r)}$ (avec $r \geq 0$) et correspond au plus proche voisin de $\mathbf{x}$ dans Λ. L'extension évite donc la saturation dans C et ramène la distorsion à l'erreur granulaire dans Λ.

2. La granularité en débit de l'extension utilisant la partition $\Lambda/2^r\Lambda$ est de 1 bit par dimension. Pour certaines applications, cette granularité peut être trop grande. On verra d'ailleurs à la section 4.5 une progression d'$1/2$ bit par dimension dans le codage dans RE_8 en utilisant plusieurs dictionnaires de base C.

3. Quand le budget de bits est limité, l'ordre d'extension r est forcément limité, si bien que le codage peut manquer de bits avant de pouvoir "capturer" le vecteur de source $\mathbf{x}$. Ce problème survient quand le budget de bits disponible pour quantifier un $\mathbf{x}$ donné est inférieur au nombre de bits nécessaires pour représenter l'indice i et le numéro de dictionnaire n codé sous forme binaire. Suivant la façon dont la quantification multidébit est utilisée, plusieurs stratégies peuvent être conçue pour gérer cette situation.

   La solution classique serait de chercher à saturer $\mathbf{x}$, par exemple au sein du code étendu d'ordre maximal pour "rentrer" dans le budget de bits. Malheureusement, à cause du codage de Voronoï, la recherche optimale (avec saturation) du plus proche voisin dans un code étendu est très ardue. Une solution plus appropriée consiste à utiliser une quantification de type gain-forme. Le vecteur de source $\mathbf{x}$ peut être réduit homothétiquement avant codage multi-débit de sorte que le plus proche voisin soit dans $C^{(r_{max})}$, où r_{max} est l'ordre le plus grand autorisé.

4. Pour un vecteur de source $\mathbf{x}$ quelconque, la complexité de l'extension telle que décrite précédemment n'est en théorie pas bornée. En effet, l'ordre d'extension r est initialisé à 0 puis incrémenter avec un pas de 1 jusqu'à ce que $\mathbf{y}$ soit dans $C^{(r)}$.

5. En pratique, l'ordre d'extension r est limité à une valeur r_{max} par la plateforme de mise en œuvre. Cette limitation est due à la taille des entiers (en nombre de bits) représentant les composantes de l'indice de Voronoï $\mathbf{k}$. Par exemple, si chaque composante est représentée sur 16 bits (non signés), on a $r_{max} = 16$.

6. En théorie, une condition importante pèse sur le dictionnaire de base C pour que l'extension de Voronoï fonctionne correctement : n'importe quel point de Λ doit pouvoir être décrit comme un mot de code dans C ou $C^{(r)}$ (avec $r = 1, 2, \cdots, +\infty$). Autrement il serait impossible d'indexer certains points de Λ. Par exemple, un dictionnaire C conçu par troncature quasi-sphérique (centrée sur l'origine) vérifie cette condition.

   En pratique, l'ordre r va de 0 à un nombre maximal r_{max} imposé par la plateforme, cette condition s'applique donc au sous-ensemble de Λ susceptible d'être utilisé.

7. L'extension de Voronoï est similaire à la technique de quantification par raffinements successifs introduite dans [18]. La technique multi-étage de [18] peut en effet être vue comme une technique de quantification multi-débit où le codage de Voronoï sert à produire des descriptions de la source de plus en plus fines après chaque étage. Cette technique possède cependant quelques limitations :

   (a) Le pas de quantification est diminué après chaque raffinement successif, et n'est donc pas adapté en cas de surcharge. Si le premier étage engendre une grande surcharge, les étages successifs ne peuvent réduire efficacement l'erreur résultante car ils sont conçus pour réduire l'erreur granulaire uniquement. La performance du premier étage est donc critique.

   (b) La propriété des raffinements successifs implique une contrainte sur les pas de quantification du codage de Voronoï et limite la performance de quantification par rapport à l'extension de Voronoï. En extension de Voronoï, le codage de Voronoï est appliqué de

façon optimale de sorte qu'un code étendu peut être vu comme un code à 2 étages où les régions de Voronoï s'emboîtent parfaitement.

4.4 Performances pour une source gaussienne : cas du codage à débits entiers par dimension dans A_2 et RE_8

La quantification vectorielle par extension de Voronoï est évaluée ici dans le cas d'une source gaussienne discrète sans mémoire (i.i.d.) de moyenne nulle et variance unité. Cette source est codée dans A_2 et RE_8 en groupant ses échantillons successifs par blocs de 2 et 8 (respectivement). On prend 10^6 vecteurs de source dans les deux cas.

Une quantification de type gain-forme est utilisée, comme montrée à la figure 4.6. Le facteur d'échelle (ou gain) g permet de régler le débit moyen de codage et contrôle la distorsion définie ici comme l'erreur quadratique moyenne. Pour un gain scalaire $g > 0$ donné, $\mathbf{x}$ est quantifié dans gA_2 en $\hat{\mathbf{x}} = g\mathbf{y}$, où $\mathbf{y}$ est le plus proche voisin de $\mathbf{x}/g$ dans A_2. Le gain g est fixé pour une quantification à débit donné ; il n'est pas quantifié. Le vecteur $\mathbf{y}$ est indexé dans A_2 ou RE_8 au moyen de l'extension de Voronoï ; cette indexation multi-débit produit un numéro de dictionnaire n (entier) et un indice i (binaire).

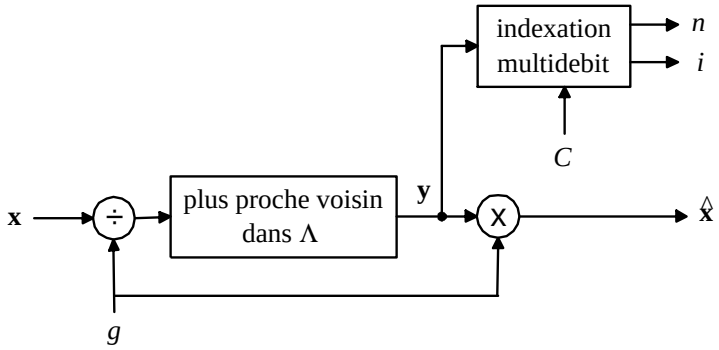


Figure 4.6: Système de quantification de type gain-forme utilisé dans la simulation.

L'indexation multi-débit de $\mathbf{y}$ implique de définir un ensemble de dictionnaires, Q_n, avec n entier (par convention $n \geq 0$). On fixe ici le débit de Q_n à n bits par dimension. Les dictionnaires de bas débit Q_0, Q_1 et Q_2 sont construits explicitement par troncature sphérique de A_2 ou RE_8, tandis que les dictionnaires de débit élevé Q_n (avec $n > 2$) sont générés de façon algorithmique par extension de Voronoï.

Pour chaque valeur de g, on mesure :

- L'erreur quadratique moyenne normalisée $D = \frac{1}{N} E\left[\|\mathbf{x} - \hat{\mathbf{x}}\|^2\right] = \frac{1}{N} E\left[\|\mathbf{x} - g\mathbf{y}\|^2\right]$;

- Le débit moyen par dimension $R = \frac{1}{N} E\left[l(n) + l(i)\right]$, où $l(n)$ et $l(i)$ correspondent respectivement au nombre de bits utilisés pour représenter n et i. Ici, $l(i) = nN$ car i est l'indice d'un mot de code dans Q_n et le débit de Q_n est de n bits par dimension. Par suite, $R = \frac{1}{N} E[l(n)] + E[n]$.

En faisant varier g, on obtient une courbe débit-distorsion $D(R)$, fonction du dictionnaire de base C.

On considère deux codages différents pour n : un codage entropique à débit minimal tel que

$$E\left[l(n)\right] = -\sum_{n \geq 0} p(n) \log_2 p(n) \tag{4.10}$$

et un codage unaire sous la forme :

n	$\rightarrow$	code unaire
0		0
1		10
2		110
3		1110
$\vdots$		$\vdots$

Dans ce dernier cas, $l(n) = n + 1$ bits et $E[l(n)] = E[n] + 1$.

4.4.1 Codage dans A_2

Les dictionnaires Q_0, Q_1 et Q_2 de 0, 1 et 2 bits par dimension sont construits explicitement à partir d'une énumération des premières orbites sphériques de A_2. Ils sont spécifiés au tableau 4.1 et illustrés à la figure 4.7 (a). Les dictionnaires Q_n de débits plus élevés ($n > 2$) sont obtenus de façon algorithmique par extension de Voronoï de Q_2. Ainsi, Q_2 joue le rôle de C, avec $R_C = 1$, et on a $Q_3 = Q_2^{(1)}$, $Q_4 = Q_2^{(2)}$, etc. On prend ici $\mathbf{a} = (0.1, 0)$ comme décalage de Voronoï. Les dictionnaires Q_3 et Q_4 sont illustrés à la figure 4.7 (b).

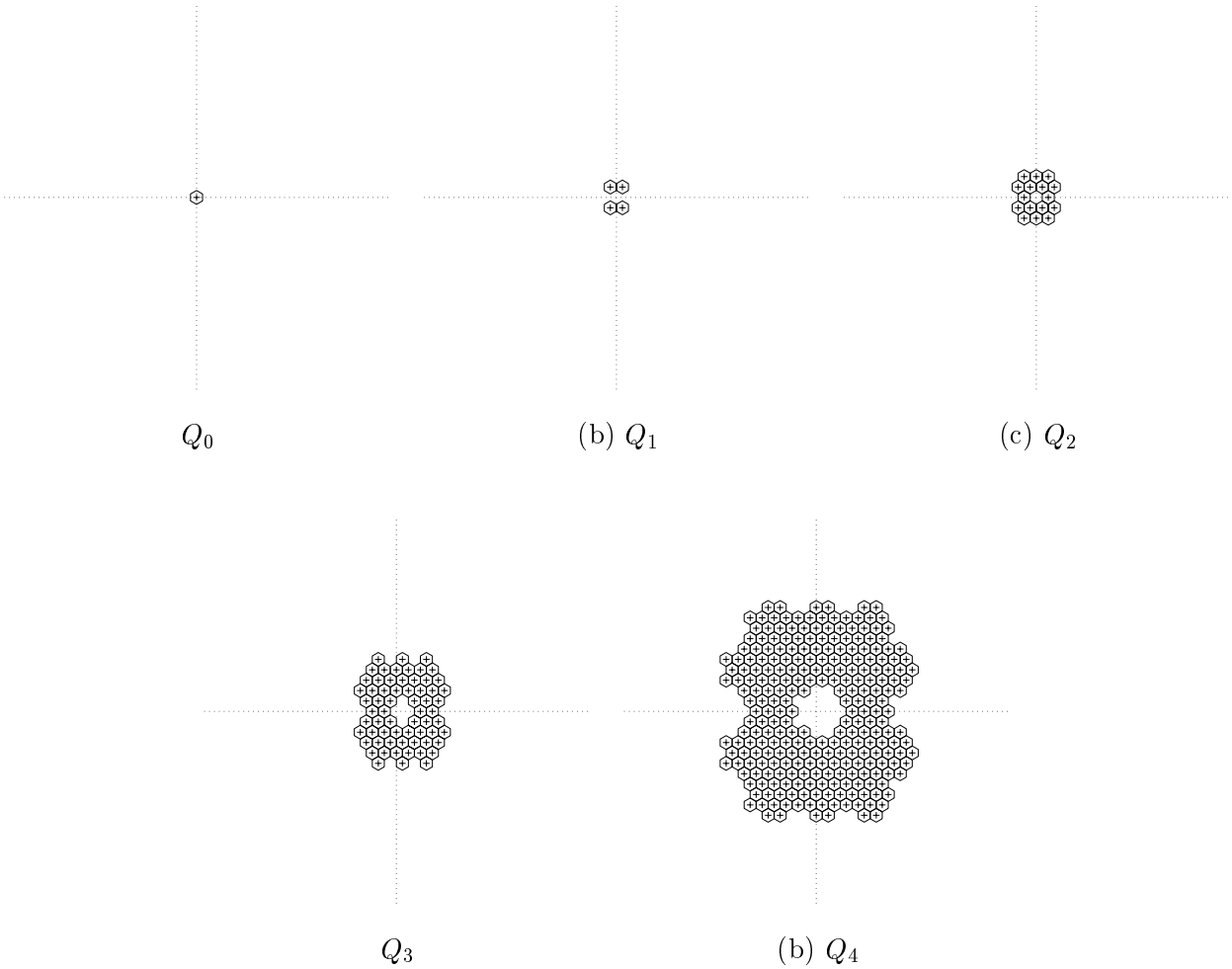
TABLEAU 4.1: Spécifications de Q_0, Q_1 et Q_2 dans A_2.

orbite	points de A_2		cardinalité	Q_0	Q_1	Q_2
0	0,	0	1	×		
1	$\pm\frac{1}{2}$,	$\pm\frac{\sqrt{3}}{2}$	4		×	×
	± 1,	0	2			×
3	0,	$\pm\sqrt{3}$	2			×
	$\pm\frac{3}{2}$,	$\pm\frac{\sqrt{3}}{2}$	4			×
4	± 1,	$\pm\sqrt{3}$	4			×

La source est représentée par une base de test de 10^6 vecteurs de dimension 2. La figure 4.8 (a) montre la probabilité $p(n)$ de n (estimée par la fréquence d'occurence de n) en fonction de g. Le nombre de bits moyen $E[l(n)]$ est également indiqué en fonction de g à la figure 4.8 (b). Quand g est suffisament grand, $\mathbf{x}/g$ devient quasi-nul et le codage est effectué dans Q_0 ; dans ce cas, $p(0) \approx 1$, $p(n) \approx 0$ pour $n > 0$ et $E[l(n)] \approx 0$.

Les performances de codage sont montrées aux figures 4.8 (c) et (d). Pour un gain g suffisamment faible, la densité de points de la source devient relativement uniforme dans chaque région de Voronoï de gA_2 ; dans ces conditions, la distorsion D est approximativement égale au moment d'ordre 2 de gA_2, soit $D \approx D_g = \frac{5}{72}g^2$. La validité de cette approximation peut être observée à la figure 4.8 (c), pour un gain g assez faible (ou de façon équivalente un débit moyen élevé).

Les courbes $R(D)$ à la figure 4.8 (d) montrent que :



Q_0 (b) Q_1 (c) Q_2

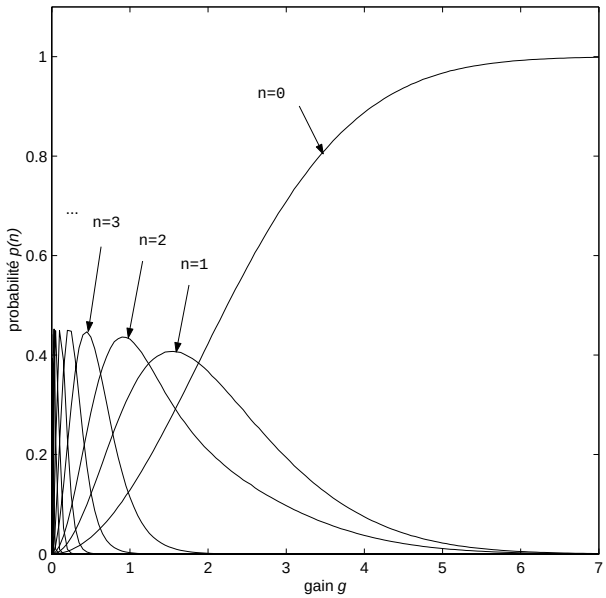
Q_3 (b) Q_4

Figure 4.7: Dictionnaires Q_n $(n = 0, 1, 2, 3, 4)$ de A_2.

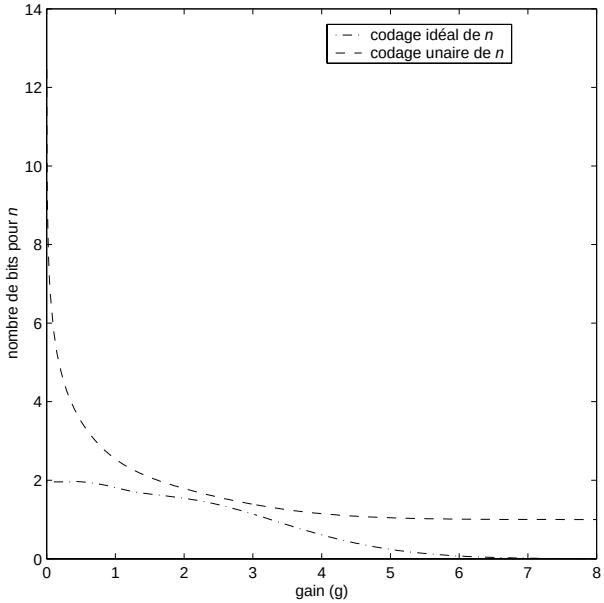
- En employant un codage entropique idéal pour n et à un débit ≥ 2 bits par dimension, l'écart entre les performances réelles et la limite de Shannon est de l'ordre de 2.5 dB ;

- Autour de 1 bit par dimension, les performances s'écartent de 3.5 dB de la limite de Shannon car le dictionnaire Q_1 est le plus probable alors que les 4 points qu'il contient ne forment pas un code sphérique efficace.

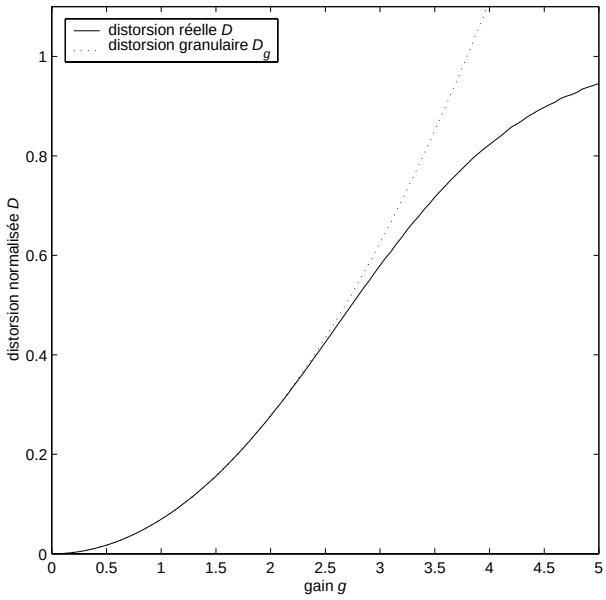
- Au dessus de 3 bits par dimension, le codage unaire n'est pas efficace à débit élevé. En fait à mesure que le débit moyen augmente, le numéro de dictionnaire n le plus probable croît et



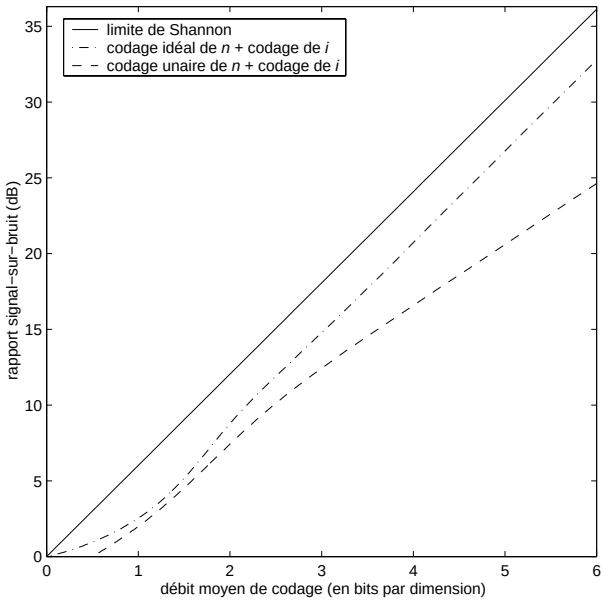
(a) probabilité de sélection



(b) $E[l(n)]$



(a) distorsion



(b) rapport signal-sur-bruit

Figure 4.8: Résultats de simulation pour la quantification multi-débit dans A_2.

donc $l(n)$ également. Au contraire, le codage de n devrait allouer le moins de bits possible pour représenter le numéro n le plus probable. Cette limitation du codage unaire est apparente (pour les faibles valeurs de g) à la figure 4.8 (b).

4.4.2 Codage dans RE_8

Les dictionnaires Q_0, Q_1 et Q_2 de 0, 1 et 2 bits par dimension sont construits explicitement à partir d'une énumération des premières orbites sphériques de RE_8. Ils sont spécifiés au tableau 4.2. Le dictionnaire Q_2 fait office de dictionnaire de base. Ainsi $Q_3 = Q_2^{(1)}$, $Q_4 = Q_2^{(2)}$, etc.

TABLEAU 4.2: Spécifications de Q_0, Q_1 et Q_2 dans RE_8.

leader absolu	cardinalité	Q_0	Q_1	Q_2
0, 0, 0, 0, 0, 0, 0, 0	1	×		
1, 1, 1, 1, 1, 1, 1, 1	128		×	×
2, 2, 0, 0, 0, 0, 0, 0	112		×	×
4, 0, 0, 0, 0, 0, 0, 0	16		×	×
2, 2, 2, 2, 0, 0, 0, 0	1120			×
3, 1, 1, 1, 1, 1, 1, 1	1024			×
4, 2, 2, 0, 0, 0, 0, 0	1344			×
2, 2, 2, 2, 2, 2, 0, 0	1792			×
3, 3, 1, 1, 1, 1, 1, 1	3584			×
4, 4, 0, 0, 0, 0, 0, 0	112			×
4, 2, 2, 2, 2, 0, 0, 0	8960			×
2, 2, 2, 2, 2, 2, 2, 2	256			×
5, 1, 1, 1, 1, 1, 1, 1	1024			×
3, 3, 3, 1, 1, 1, 1, 1	7168			×
6, 2, 0, 0, 0, 0, 0, 0	224			×
4, 4, 2, 2, 0, 0, 0, 0	6720			×
4, 2, 2, 2, 2, 2, 2, 0	7168			×
5, 3, 1, 1, 1, 1, 1, 1	7168			×
3, 3, 3, 3, 1, 1, 1, 1	8960			×
4, 4, 4, 0, 0, 0, 0, 0	448			×
3, 3, 3, 3, 3, 1, 1, 1	7168			×
7, 1, 1, 1, 1, 1, 1, 1	1024			×
8, 0, 0, 0, 0, 0, 0, 0	16			×

Les résultats sont illustrés à la figure 4.9. Par rapport au codage dans A_2, la probabilité estimée $p(n)$ de sélection de Q_n, à la figure 4.9 (a), est plus localisée pour un n donné et atteint un maximum

plus élevé. En effet, en dimension 8, la source gaussienne tend à se concentrer sur une coquille sphérique (propriété de *sphere hardening*) ; en outre, les dictionnaires Q_n dans RE_8 sont plus sphériques que dans A_2. La figure 4.9 (b) permet de vérifier la sous-optimalité du codage unaire.

La distorsion moyenne D peut être approximée pour un gain suffisamment faible par le moment d'ordre 2 du réseau dilaté gRE_8 ; l'approximation $D \approx D_g = 4G(RE_8)g^2$ avec $G(RE_8) \approx 0.071682$ est validée à la figure 4.9 (c).

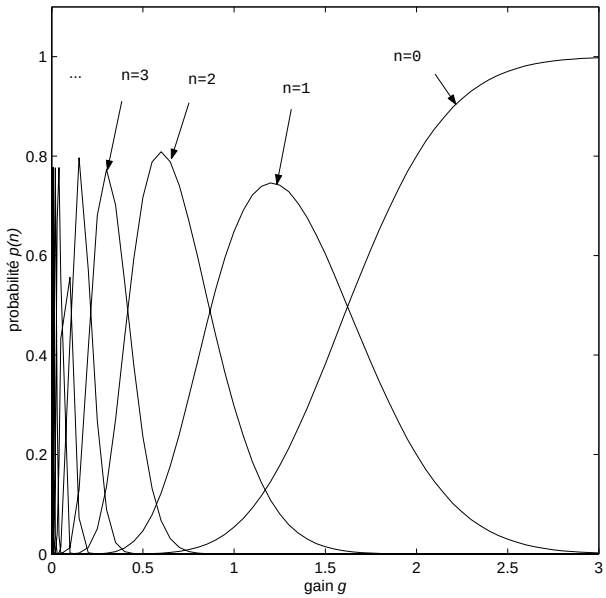
Les performances montrées à la figure 4.9 (d) sont meilleures que celles du codage dans A_2. En effet, l'information supplémentaire n est associée à un bloc de 8 échantillons et non de 2 ; son débit relatif (en bits par dimension) est donc réduit. De plus, d'après l'allure de la probabilité $p(n)$ de sélection de Q_n, pour un gain g fixé la distribution de $p(n)$ en fonction de n est très inégale ; le débit de codage entropique de n est plus faible en codage dans RE_8 qu'en codage dans A_2. Enfin, le réseau RE_8 apporte un gain granulaire par rapport à A_2.

Avec un codage entropique de n, l'écart entre les performances mesurées et la limite de Shannon est de l'ordre de 2 dB.

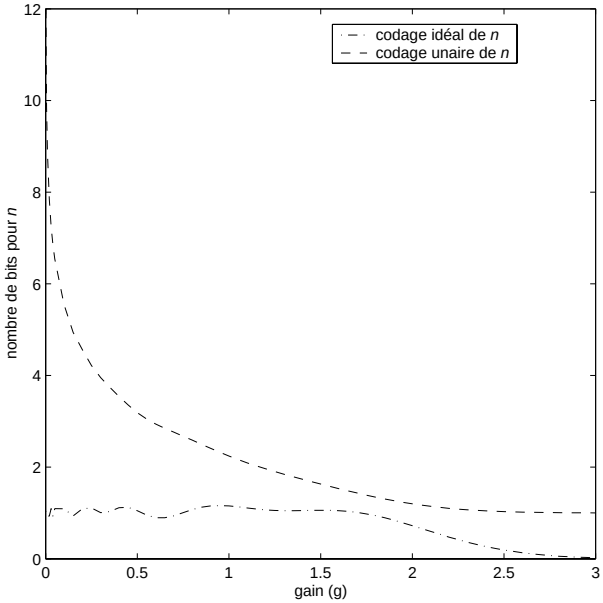
4.5 Application : quantification vectorielle algébrique multi-débit basée sur le réseau RE_8 pour le codage TCX audio

On présente ici une application de l'extension de Voronoï au codage par transformée de la cible TCX. Cette application est en fait développée pour le codage TCX sans prédiction de pitch des signaux de parole et de musique tel qu'utilisé en codage ACELP/TCX multi-mode [5]. L'extension de Voronoï est utilisée afin d'éliminer les limites de [1]. La quantification multi-débit peut être vue comme une extension de la technique de [1].

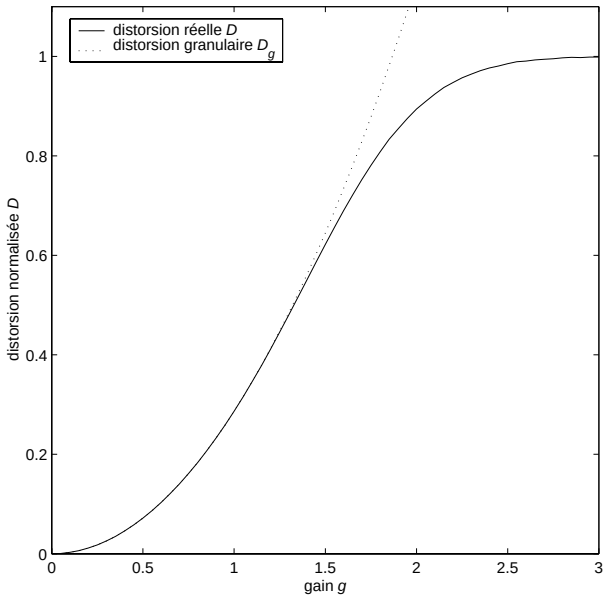
La cible TCX transformée est découpée en sous-vecteurs de dimension 8 comme dans [1]. Chaque sous-vecteur est quantifié dans le réseau RE_8. Le débit possible de codage dans RE_8 évolue par incrément de 1/2 bit par dimension, soit 4 bits en dimension 8.



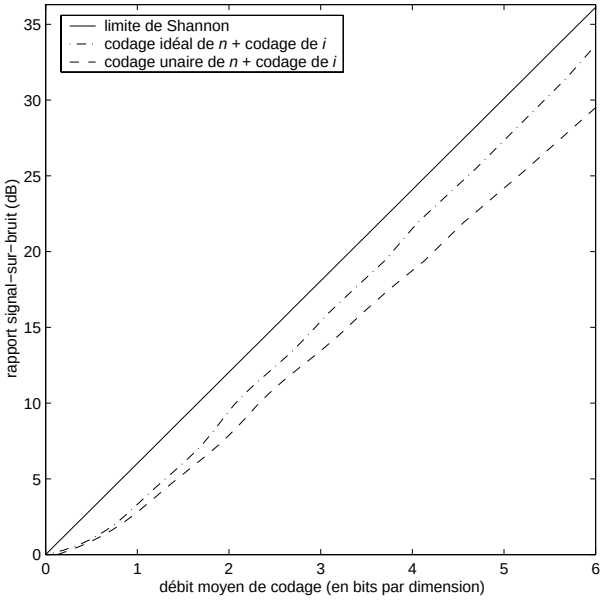
(a) probabilité de sélection



(b) $E[l(n)]$



(c) distorsion



(d) rapport signal-sur-bruit

Figure 4.9: Résultats de simulation pour la quantification multi-débit dans RE_8.

4.5.1 Quantification multi-débit

Le système de quantification multi-débit, illustré à la figure 4.10, comprend un codeur et un décodeur. Le sous-vecteur $\mathbf{x}$ est de dimension 8 ; le codage génère un indice i (binaire) et un numéro de dictionnaire n (entier), comme la technique de [1]. Le numéro de dictionnaire n identifie un code issu de RE_8 noté Q_n. Contrairement à [1], chaque Q_n est un sous-ensemble strict de RE_8. Mais comme dans [1], le débit de Q_n est $4n/8 = n/2$ bits par dimension. L'indice i est donc représenté sur $4n$ bits par sous-vecteur de dimension 8. Le décodeur utilise les mêmes dictionnaires Q_n que le codeur ; si le budget de bit n'est pas limité, il reconstruit simplement le point $\mathbf{y}$ indexé par n et i.

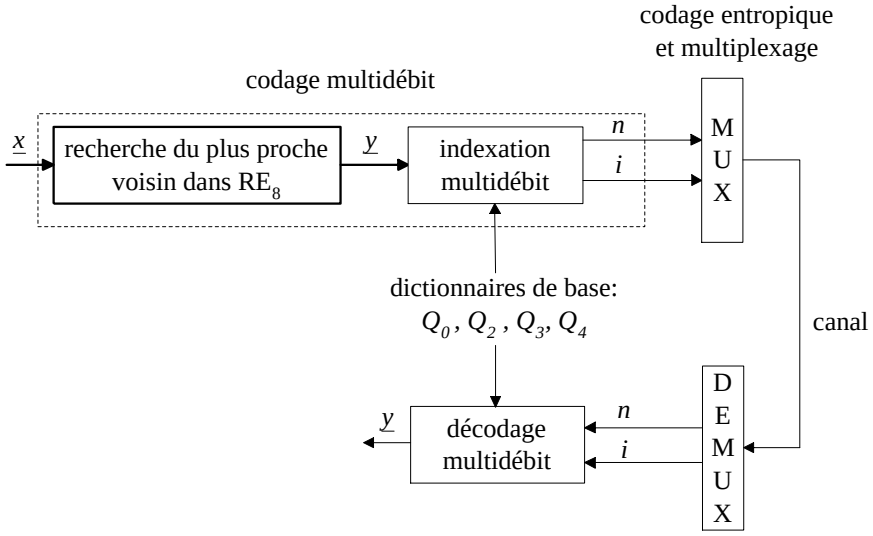


Figure 4.10: Système de quantification multi-débit basé sur RE_8.

Le numéro de quantificateur n est défini ici dans l'ensemble $\{0, 2, 3, 4, 5, 6, \cdots, 36\}$. Le cas $n = 1$ est interdit car il correspond à une allocation de 4 bits en dimension 8. À un tel bas débit, la quantification par réseau de points n'est pas efficace, comparativement à une simple injection de bruit. Dans [1], n est restreint à l'ensemble $\{0, 1, 2, 3, 4, 5\}$. Ainsi le système décrit ici permet une allocation des bits plus flexible, tout en interdisant le cas $n = 1$ pour plus d'efficacité. La limitation à $n = 36$ est justifiée en supposant que les composantes des indices de Voronoï sont représentées par des entiers de 16 bits (non signés).

Dictionnaires Q_n : Les dictionnaires de quantification sont divisés en 2 catégories :

1) Les dictionnaires de base à bas débit Q_0, Q_2, Q_3 et Q_4 qui sont des constellations quasi-sphériques classiques. Ces dictionnaires sont décrits physiquement par des tables de tailles réduites associées à des vecteurs directeurs (ou leaders) [19, 1]. Dans le système décrit ici, les codes Q_2 et Q_3 sont imbriqués, c'est-à-dire que $Q_2 \subset Q_3$; cette contrainte est imposée pour le bon fonctionnement de l'extension de Voronoï.

   Les leaders absolus définissant Q_0, Q_2, Q_3 et Q_4 sont donnés dans le tableau 4.3. On peut vérifier que le dictionnaire Q_2 est identique au Q_1 défini dans le tableau 4.2. Le code Q_4 contient $2^{16} - 16$ points de RE_8 ; le leader absolu $(20, 0, \cdots, 0)$ pourrait le compléter mais l'extension de Voronoï telle qu'optimisée ici ne serait plus applicable. Les contraintes de signes associés aux leaders absolus ne sont pas mentionnées car elles se déduisent des propriétés de RE_8 [19].

2) Les dictionnaires de débit élevé Q_n, avec $n > 4$, sont construits virtuellement en appliquant l'extension de Voronoï alternativement sur Q_3 et Q_4. Le pas dans l'allocation des bits est donc de $\frac{1}{2}$ bit par dimension. Ainsi, Q_5 est généré par extension d'ordre 1 de Q_3, Q_6 par extension d'ordre 1 de Q_4, Q_7 par extension d'ordre 2 de Q_3, etc. Plus généralement, $Q_{2r+3} = Q_3^{(r)}$ et $Q_{2r+4} = Q_4^{(r)}$ où $r \geq 0$ est l'ordre d'extension de Voronoï. Le décalage de Voronoï $\mathbf{a}$ est fixé à $\mathbf{a} = (2, 0, 0, 0, 0, 0, 0, 0)$ – il s'agit d'un trou profond de RE_8. C'est un décalage non-admissible. On peut en effet lever la contrainte d'admissibilité sur $\mathbf{a}$ si l'extension de Voronoï utilise les algorithmes d'indexation de Voronoï $\mathbf{k} \rightarrow \mathbf{y}$ et $\mathbf{y} \rightarrow \mathbf{k}$.

4.5.2 Codage multi-débit

On suppose ici qu'on cherche à quantifier un sous-vecteur $\mathbf{x}$ de dimension 8 de la cible TCX. Le codage se réduit à une recherche du plus proche voisin et à une indexation multi-débit employant l'extension de Voronoï. Cette indexation fonctionne par définition pour un dictionnaire de base unique C en itérant l'ordre d'extension r jusqu'à ce que $\mathbf{y}$ soit dans $C^{(r)}$; de la sorte, la complexité

TABLEAU 4.3: Spécifications de Q_0, Q_2, Q_3 et Q_4 dans RE_8.

leader absolu	cardinalité	Q_0	Q_2	Q_3	Q4	s	k_a
0, 0, 0, 0, 0, 0, 0, 0	1	×				−	36
1, 1, 1, 1, 1, 1, 1, 1	128		×	×		1	0
2, 2, 0, 0, 0, 0, 0, 0	112		×	×		4	1
2, 2, 2, 2, 0, 0, 0, 0	1120			×		8	2
3, 1, 1, 1, 1, 1, 1, 1	1024			×		11	3
4, 0, 0, 0, 0, 0, 0, 0	16		×	×		32	4
2, 2, 2, 2, 2, 2, 0, 0	1792				×	12	5
3, 3, 1, 1, 1, 1, 1, 1	3584				×	21	6
4, 2, 2, 0, 0, 0, 0, 0	1344			×		36	7
2, 2, 2, 2, 2, 2, 2, 2	256				×	16	8
3, 3, 3, 1, 1, 1, 1, 1	7168				×	31	9
4, 2, 2, 2, 2, 0, 0, 0	8960				×	40	10
4, 4, 0, 0, 0, 0, 0, 0	112			×		64	11
5, 1, 1, 1, 1, 1, 1, 1	1024				×	79	12
3, 3, 3, 3, 1, 1, 1, 1	8960				×	41	13
4, 2, 2, 2, 2, 2, 2, 0	7168				×	44	14
4, 4, 2, 2, 0, 0, 0, 0	6720				×	68	15
5, 3, 1, 1, 1, 1, 1, 1	7168				×	89	16
6, 2, 0, 0, 0, 0, 0, 0	224			×		164	17
4, 4, 4, 0, 0, 0, 0, 0	448				×	96	18
6, 2, 2, 2, 0, 0, 0, 0	4480				×	168	19
6, 4, 2, 0, 0, 0, 0, 0	2688				×	196	20
7, 1, 1, 1, 1, 1, 1, 1	1024				×	301	21
8, 0, 0, 0, 0, 0, 0, 0	16			×		512	22
6, 6, 0, 0, 0, 0, 0, 0	112				×	324	23
8, 2, 2, 0, 0, 0, 0, 0	1344				×	516	24
8, 4, 0, 0, 0, 0, 0, 0	224				×	544	25
9, 1, 1, 1, 1, 1, 1, 1	1024				×	821	26
10, 2, 0, 0, 0, 0, 0, 0	224				×	1252	27
8, 8, 0, 0, 0, 0, 0, 0	112				×	1024	28
10, 6, 0, 0, 0, 0, 0, 0	224				×	1412	29
12, 0, 0, 0, 0, 0, 0, 0	16				×	2592	30
12, 4, 0, 0, 0, 0, 0, 0	224				×	2624	31
10, 10, 0, 0, 0, 0, 0, 0	112				×	2500	32
14, 2, 0, 0, 0, 0, 0, 0	224				×	4804	33
12, 8, 0, 0, 0, 0, 0, 0	224				×	3104	34
16, 0, 0, 0, 0, 0, 0, 0	16				×	8192	35

de l'indexation n'est pas bornée. Pour majorer cette complexité, le codage décrit ici utilise au maximum 2 itérations. De plus, on prend en compte le fait que deux dictionnaires de base Q_3 et Q_4 sont étendus au lieu d'un seul.

Les étapes de codage sont :

1. Trouver le proche voisin $\mathbf{y}$ de $\mathbf{x}$ dans le réseau infini RE_8.

2. Vérifier si $\mathbf{y}$ est un élément de Q_0, Q_2, Q_3 ou Q_4 au moyen de l'algorithme détaillé à l'annexe 4.A. Cette vérification produit un numéro de dictionnaire $n \in \{0, 2, 3, 4, out\}$, où out est un entier quelconque, et un identificateur de leader absolu k_a si $n > 0$. La valeur out peut être fixée par exemple à $out = 100$; elle est utilisée pour indiquer une détection de surcharge. Autrement dit, si $n = out$, $\mathbf{y}$ n'est dans aucun des dictionnaires de base.

3. Il y a 2 cas :

   - Cas $n \in \{0, 2, 3, 4\}$:
     Si $n = 0$, arrêter.
     Si $n = 2, 3$ ou 4, indexer le vecteur $\mathbf{y}$ dans Q_n à partir de k_a puis arrêter. L'indexation de $\mathbf{y}$ est basée sur le calcul du rang de la permutation et des consultations dans des tables comme dans [19, 20, 21, 22].
   - Cas $n = out$: Appliquer l'extension de Voronoï sur Q_3 et Q_4 en 2 itérations. Cette opération est détaillée à l'annexe 4.B pour ne pas alourdir le texte.

4.5.3 Décodage multi-débit

On décode n à partir de sa représentation binaire. L'indice i est interprété différemment suivant le numéro de dictionnaire n :

1) Si $n = 0$, le décodeur reconstruit $\mathbf{y} = (0, \cdots, 0)$, le seul élément du dictionnaire de base Q_0.

2) Si $n \in \{2, 3, 4\}$, le décodeur démultiplexe l'indice i de taille $4n$ bits à partir du canal, puis décode i comme un indice identifiant un mot de code $\mathbf{y}$ dans un dictionnaire Q_2, Q_3 ou Q_4 (suivant n). Le décodage de i peut être réalisé par une technique de [19, 20, 21, 22].

3) Si $n > 4$, l'extension de Voronoï est utilisée. Le décodeur démultiplexe l'indice i de taille $4n$ bits comme un indice j et un indice $\mathbf{k}$. L'ordre d'extension r est déduit de n en calculant le quotient de $(n-3)/2$; le modulo de Voronoï m est alors calculé : $m = 2^r$. Ensuite la quantité $2r$ est soustraite de n pour identifier Q_3 ou Q_4 et l'indice j est décodé en $\mathbf{c}$ sur la base de la valeur modifiée de n. Le décodage de j est basé sur le décodage du rang et des consultations dans des tables comme dans [19, 20, 21, 22]. En utilisant l'algorithme de [14] repris au chapitre 3, l'indice de Voronoï $\mathbf{k}$ est décodé en $\mathbf{v}$ en cherchant le plus proche voisin $\mathbf{z}$ de $(\mathbf{k}M(RE_8) - \mathbf{a})/m$ dans RE_8, et calculant $\mathbf{v} = \mathbf{k}M(RE_8) - m\mathbf{z}$. Finalement, le décodeur reconstruit $\mathbf{y} = m\mathbf{c} + \mathbf{v}$.

4.5.4 Complexité de quantification

La complexité de la quantification multi-débit dans RE_8 a été évaluée dans [23] pour une implémentation sur processeur spécialisé (TMS320C54x). Pour le codage TCX avec des trames de 20 ms de signaux échantillonnés à 16 kHz et décimés à 12.8 kHz, le codage multi-débit de 32 sous-vecteurs de dimension 8 requiert environ 15 MIPS (millions d'intructions par seconde) et le décodage multi-débit 5 MIPS. La complexité est due essentiellement à l'indexation multi-débit (calcul de n et de i). Les résultats de [23] sont résumés au tableau 4.4 ; la complexité en MIPS s'obtient en multipliant le nombre de cycles par le nombre d'exécutions par seconde – ici le codage et le décodage sont exécutés 1600 fois par seconde (32 sous-vecteurs codés par trame, 50 trames par seconde).

TABLEAU 4.4: Complexité de la quantification multi-débit dans RE_8.

(a) Codage multi-débit

Bloc	Nombre de cycles (estimés)
Décodage de RE_8	900
dont: - décodage de D_8	300
Calcul de n et k_a	7000
dont: - calcul $\mathbf{k} \rightarrow \mathbf{y}$	1800
- calcul $\mathbf{y} \rightarrow \mathbf{k}$	100
Calcul de i à partir de n et k_a	2000
dont: - calcul du rang de la permutation	1500
Total	9900

(b) Décodage multi-débit

Bloc	Nombre de cycles (estimés)
Décodage de n et i	3300
dont: - décodage du rang de la permutation	900
- décodage de i	1400
- calcul $\mathbf{k} \rightarrow \mathbf{y}$	1800
Total	3300

4.6 Généralisations

4.6.1 Extension par représentants de cosets de $\Lambda/2^r\Lambda$

L'extension d'un code C peut être définie plus généralement comme :

$$C^{(r)} = 2^r C + [\Lambda/2^r\Lambda] = \bigcup_{\substack{\mathbf{c}\in C \\ \mathbf{v}\in[\Lambda/2^r\Lambda]}} (2^r\mathbf{c} + \mathbf{v})\,. \tag{4.11}$$

où $[\Lambda/2^r\Lambda]$ est un ensemble de représentants de cosets pour la partition $\Lambda/2^r\Lambda$. Le partionnement de réseau de points Λ induit par un sous-réseau est revu au chapitre 2. Au lieu de prendre un code de Voronoï $V(\Lambda, 2^r\Lambda)$, on peut par exemple choisir :

$$[\Lambda/2^r\Lambda] = \left\{\mathbf{k}M(\Lambda) | \mathbf{k} \in \mathbb{Z}^N \text{ et } -2^{r-1} \leq k_i < 2^{r-1} \text{ pour } 1 \leq i \leq N\right\} \tag{4.12}$$

où $M(\Lambda)$ est une matrice génératrice de Λ (idéalement réduite).

4.6.2 Extension de Voronoï à partir d'une partition Λ/Λ' où Λ' est un sous-réseau de Λ

L'extension de Voronoï peut être aussi généralisée en utilisant des codes de Voronoï $V(\Lambda, \Lambda')$ définis dans [4] :

$$V(\Lambda, \Lambda') = \Lambda \cap \left\{ V(\Lambda') + \mathbf{a} \right\}, \tag{4.13}$$

où le décalage $\mathbf{a} \in \mathbb{R}^N$ définissant $V(\Lambda, \Lambda')$ est identique pour chaque Λ'. Dans ce cas, $V(\Lambda, \Lambda')$ est généré en tronquant Λ par une région de Voronoï d'un sous-réseau Λ'. L'extension de Voronoï $C^{(r)}$ de C d'ordre r, où $r = 1/N \log_2 |\Lambda/\Lambda'|$ est en général non-entier. Elle est alors définie par :

$$C^{(r)} = T_{\Lambda \to \Lambda'} C + V(\Lambda, \Lambda'), \tag{4.14}$$

où $T_{\Lambda \to \Lambda'}$ désigne une matrice de passage de Λ à Λ'. Le débit du code résultant est de $R_C + r$ bits par dimension, où R_C est le débit par dimension de C. Si $\Lambda' = m\Lambda$, où m est un entier ≥ 2, l'ordre d'extension est $r = \log_2 m$. Cette généralisation englobe les cas où $\Lambda = R\Lambda$ avec $|\Lambda/R\Lambda| = 2^{N/2}$ et R est une matrice de rotation (par exemple une rotation de Hadamard si N est pair).

Les algorithmes de quantification par extension de Voronoï peuvent être généralisés au cas d'une partition Λ/Λ'.

4.7 Conclusions

Ce chapitre a présenté une méthode algorithmique – l'extension de Voronoï – permettant d'étendre des constellations issues de réseaux de points et de construire ainsi efficacement des quantificateurs vectoriels algébriques multi-débits adaptés au codage par transformée. D'après les simulations réalisées avec une source gaussienne i.i.d., les performances de la quantification vectorielle par troncature sphérique et extension de Voronoï sont proches de la limite de Shannon si la dimension N est suffisamment élevée et si l'information supplémentaire est représentée par codage entropique.

Une application concrète, développée dans le contexte du codage multi-mode ACELP/TCX, a

été étudiée. L'extension de Voronoï a ainsi été utilisée pour concevoir un système de quantification algébrique multi-débit en dimension 8, basé sur le réseau RE_8 et étendant le système de [1] pour le codage TCX audio. Ce système met en œuvre une forme optimisée d'extension de Voronoï, où le nombre d'itérations sur l'ordre d'extension r est limité à 2. La quantification multi-débit s'appuie sur un ensemble de quelques dictionnaires algébriques de tailles réduites ; l'extension de Voronoï est utilisée pour générer des dictionnaires de débits plus élévés.

La quantification algébrique multi-débit conçue à partir de l'extension de Voronoï possède les avantages suivants pour le codage par transformée de la cible TCX :

- Faible stockage : L'extension de Voronoï étant algorithmique, les codes étendus $C^{(r)}$ – à haut débit – sont générés virtuellement sans stockage. On peut donc ainsi allouer plus de bits qu'avec des techniques vectorielles classiques, en dimension élevée. Le stockage se réduit essentiellement aux codes algébriques C à faible débit.

- Débit virtuellement illimité : En théorie, l'ordre d'extension de Voronoï r – donc le débit de quantification – est illimité ; cet ordre est en pratique limité par la résolution des indices de Voronoï – par exemple, r est limité à 16, si les composantes des indices de Voronoï sont représentés sur 16 bits.

- Flexibilité : Les codes $C^{(r)}$ générés par extension dépendent des codes de taille fixe C sur lesquels l'extension est appliquée ; cette famille $C^{(r)}$ peut donc être optimisée pour la source (en optimisant C par exemple par sélection statistique de leaders absolus).

Les inconvénients de l'extension de Voronoï ne doivent pas pour autant être occultés :

- Allocation implicite des bits : Le codage multi-débit développé ici choisit lui-même une allocation des bits adaptative pour éviter la surcharge dans les dictionnaires de base. Il est donc impossible d'imposer directement un certain budget de bits comme en codage par transformée classique [24].

- Surcharge interdite : La quantification par extension de Voronoï n'est pas conçue pour gérer des cas de surcharge dans les dictionnaires étendus. Cette surcharge peut être en général évitée car la dynamique des signaux réels est limitée – la résolution des indices de Voronoï doit alors être suffisante pour représenter cette dynamique.

- Granularité en débit : L'extension de Voronoï à partir d'une partition $\Lambda/2^r\Lambda$ conduit à une granularité fixe de 1 bit par dimension. On peut néanmoins obtenir des incréments fractionnaires de débit en utilisant plusieurs dictionnaires de base.

- Nombre d'itérations : L'extension de Voronoï est réalisée par une boucle sur l'ordre r. Afin de majorer la complexité de calcul due à l'extension, la valeur initiale de r et l'adaptation de r doivent être bien choisies pour limiter la boucle de codage à quelques itérations ; ces procédures (initialisation et adaptation de r) doivent être conçues au cas par cas.

- Contrainte structurelle : Les codes à haut débit $C^{(r)}$ sont déterminés par C.

- Information supplémentaire non bornée : En pratique il est difficile d'obtenir des statistiques valables sur la probabilité de sélection des dictionnaires en quantification multi-débit par extension de Voronoï.

L'extension de Voronoï est fondée sur la relation fondamentale de décomposition en cosets :

$$\Lambda = m\Lambda + V(\Lambda, m\Lambda), \tag{4.15}$$

qui met en évidence la propriété d'auto-similarité des réseaux de points. Des réseaux auto-similaires ont d'ailleurs été utilisés dans [25]. On pourrait par la suite explorer le lien entre l'extension de Voronoï, la décomposition dyadique par ondelettes et les fractales. Des problèmes, comme le codage sans perte efficace des numéros de dictionnaire n et l'analyse des performances asymptotiques de quantification $D(R)$ pour une source sans mémoire peuvent également être étudiés.

Annexe 4.A : Algorithme d'identification de leader absolu et de détection de surcharge

L'approche classique pour vérifier si un point $\mathbf{y}$ est dans un dictionnaire algébrique spécifié par des leaders absolus consiste [19] à calculer le leader absolu associé à $\mathbf{y}$, en triant par ordre décroissant le vecteur $(|y_1|, \cdots, |y_8|)$, puis à vérifier si ce leader fait partie d'une table comprenant la liste des leaders absolus définissant le dictionnaire. Un algorithme plus rapide spécifique à RE_8 peut aussi être employé en calculant un identificateur de leaders absolus à partir de $\mathbf{y}$. L'algorithme est décrit en 3 étapes :

(a) Calculer $s = (y_1^4 + \cdots + y_8^4)/8$ à partir de $\mathbf{y}$. Les permutations signées de $\mathbf{y}$ donnent toutes la même valeur. On peut montrer que s identifie de façon unique un leader absolu dans le tableau 4.3. Les valeurs de s associées aux leaders absolus de ce tableau (exceptés le mot nul) sont pré-calculées et stockées dans un vecteur pré-défini indexé par $0 \leq k_a \leq 35$.

(b) Chercher s dans ce vecteur prédéfini. Si s n'est pas dans ce vecteur, on a donc une surcharge. L'indice k_a résultant est compris entre 0 et 35. Si $s = 0$, $\mathbf{y} = 0$ et on fixe $k_a = 36$. Si s n'est pas nul et n'est dans le vecteur prédéfini, on fixe $k_a = 37$ (pour indiquer une surcharge).

Le numéro de dictionnaire n peut alors être trouvé par consultation dans une table adéquate indexée par k_a.

Annexe 4.B : Extension de Voronoï en 2 itérations

La saturation est gérée par extension de Voronoï. L'extension est limitée à 2 itérations en initialisant adéquatement l'ordre d'extension r.

Les étapes sont :

(i) Initialisation : Pré-sélectionner l'ordre d'extension r et le modulo de Voronoï m à partir de $\mathbf{y} = (y_1, \cdots, y_8)$ afin de minimiser le nombre d'itérations sur r. Cette pré-sélection est détaillée plus loin. Initialiser le compteur $iter = 0$.

(ii) Si $iter = 2$, aller à l'étape (vii).

(iii) Trouver le plus proche voisin $\mathbf{c}$ de $(\mathbf{y} - \mathbf{a})/m$ dans RE_8.

(iv) Vérifier si $\mathbf{c}$ est un élément de Q_2, Q_3 ou Q_4 – les détails sont donnés à l'annexe 4.A. Cette vérification produit un numéro de dictionnaire n et un identificateur k_a de leader absolu. A ce stade, $\mathbf{c}$ ne peut pas être dans Q_0. Par conséquent, n peut prendre ses valeurs dans $\{2, 3, 4, out\}$.

(v) Il y a deux cas :

- Si $n = out$, incrementer l'ordre d'extension r par 1 et multiplier le facteur d'échelle m par 2.
- Si $n = 2$, fixer $n = 3$. En effet, si $\mathbf{c}$ est dans Q_2, $\mathbf{c}$ est aussi dans Q_3 car Q_2 est imbriqué dans Q_3 ($Q2 \subset Q3$). Si $n = 3$ ou 4, l'ordre d'extension r et le facteur $m = 2^r$ sont suffisants pour quantifier $\mathbf{x}$ sans saturation.

  Calculer le vrai numéro de dictionnaire : $n^* = n + 2r$. Fixer $\mathbf{c}^* = \mathbf{c}$, $k_a^* = k_a$, $n^* = n$, $m^* = m$. Décrémenter l'ordre d'extension r de 1 et diviser m par 2 – la pré-sélection peut en effet sur-estimer l'ordre r suffisant pour capturer $\mathbf{y}$ et donc gaspiller des bits.

(vi) Incrémenter le compteur d'itérations $iter$ par 1. Aller à l'étape (ii).

(vii) Fixer $n = n^*$. Calculer l'indice j de $\mathbf{c}^*$ à partir de k_a^* et l'indice de Voronoï $\mathbf{k}$ de $\mathbf{v} = \mathbf{y} - m^*\mathbf{c}^*$. L'indexation de $\mathbf{c}^*$ est basée sur le codage du rang et des consultations dans des tables comme dans [19, 20, 21, 22]. Dans le cas particulier de RE_8, l'indice $\mathbf{k}$ correspond au vecteur $\mathbf{v}M^{-1}(RE_8)$ pris modulo m^* sur chaque composante, où $M(RE_8)$ est une matrice génératrice de RE_8. Multiplexer ces deux indices, j et $\mathbf{k}$, pour former i.

La pré-sélection de r et m est réalisée comme suit. On calcule d'abord $t = (y_1^2 + \cdots + y_8^2)/32$. On fixe $r = 1$ et $m = 2^r = 2$, puis tant que $t > 11$, calculer $t = t/4$, $r = r + 1$ et $m = 2m$. Alternativement on peut calculer r et m sans boucle à partir de t. Cette procédure de pré-sélection est motivée par 2 observations :

- En passant de l'ordre d'extension r à $r + 1$, les dictionnaires de base Q_3 et Q_4 sont multipliés par $m = 2^{r+1}$ au lieu de 2^r. La norme quadratique des mots de code dans Q_3 et Q_4 est donc multipliée par 4. Ceci explique le facteur $1/4$ appliqué à t à chaque itération.

- L'union des dictionnaires de base $Q0 \cup Q2 \cup Q3 \cup Q4$ comprend des points de RE_8 sur les orbites 0 à 32. La dernière orbite complète est l'orbite 5. La constante fixant la fin de boucle quand $t > 11$ est sélectionnée expérimentalement entre 5 et 32.

BIBLIOGRAPHIE

[1] M. Xie and J.-P. Adoul, "Embedded algebraic vector quantization (EAVQ) with application to wideband audio coding," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Atlanta, GA, U.S.A, 1996, vol. 1, pp. 240–243.

[2] G.D. Forney, "Coset codes. I. Introduction and geometrical classification," *IEEE Trans. Inform. Theory*, vol. 34, no. 5, pp. 1123–1151, Sep. 1988.

[3] G.D. Forney and L.-F Wei, "Multidimensional Constellations - Part I: Introduction, Figures of Merit, and Generalized Cross Constellations," *IEEE J. Select. Areas Com.*, vol. 7, no. 6, pp. 877–892, Aug. 1989.

[4] G.D. Forney, "Multidimensional Constellations - Part II: Voronoi Constellations," *IEEE Trans. Selected Areas in Communications*, vol. 7, no. 6, pp. 941–958, Aug. 1989.

[5] B. Bessette, R. Salami, C. Laflamme, and R. Lefebvre, "A wideband speech and audio codec at 16/24/32 kbit/s unsing hybrid ACELP/TCX techniques," in *Proc. IEEE Workshop on Speech Coding*, June 1999, pp. 7–9.

[6] J.-P. Adoul and R. Lefebvre, *Wideband Speech Coding*, chapter 8, pp. 289–309, Elsevier (W.B. Kleijn and K.K. Paliwal, eds.), 1995.

[7] V. Sanchez and J.-P. Adoul, "Low-delay wideband speech coding using a new frequency domain approach," in *Proc. ICASSP*, 1993, vol. 2, pp. 415–418.

[8] R. Lefebvre, R. Salami, C. Laflamme, and J.-P. Adoul, "8 kbit/s coding of speech with 6 ms frame-length," in *Proc. ICASSP*, 1993, vol. 2, pp. 612–615.

[9] R. Lefebvre, R. Salami, C. Laflamme, and J.-P. Adoul, "High quality of wideband audio signals using transform coded excitation (TCX)," in *Proc. ICASSP*, 1994, vol. I, pp. 193–196.

[10] J.-M. LeRoux, R. Lefebvre, and J.-P. Adoul, "Comparison of the wavelet decomposition and the Fourier transform in TCX encoding of wideband speech and audio," in *Proc. ICASSP*, 1995, vol. 5, pp. 3083–3086.

[11] J.-H. Chen and D. Wang, "Transform predictive coding of wideband speech signals," in *Proc. ICASSP*, May 1996, vol. 1, pp. 275–278.

[12] J. Schnitzler, J. Eggers, C. Erdmann, and Peter Vary, "Wideband speech coding using forward/backward adaptive prediction with mixed time/frequency domain excitation," in *Proc. IEEE Workshop on Speech Coding*, 1999, pp. 4–6.

[13] S.A. Ramprashad, "A multimode transform predictive coder (MPTC) for speech and audio," in *Proc. IEEE Workshop on Speech Coding*, 1999, pp. 10–12.

[14] J.H. Conway and N.J.A. Sloane, "A fast encoding method for lattice codes and quantizers," *IEEE Trans. on Inform. Theory*, vol. 29, no. 6, pp. 820–824, Nov. 1983.

[15] N.S. Jayant and P. Noll, *Digital Coding of Waveforms – Principles and Applications to Speech and Video*, Prentice-Hall, 1984.

[16] S.N. Diggavi, N.J.A. Sloane, and V.A. Vaishampayan, "Design of Asymetric Multiple Description Lattice Vector Quantizers," in *Proceedings DCC 2000: Data Compression Conference (Snowbird, 2000), J. A. Storer and M. Cohn (editors), IEEE Computer Society, Los Alamitos, CA*, 2000, pp. 490–499.

[17] T. Moriya and H. Suda, "An 8 kbit/s transform coder for noisy channels," in *International Conference on Acoustics, Speech, and Signal Processing*, 1989, vol. 1, pp. 196–199.

[18] D. Mukherjee and S.K. Mitra, "Vector set-partiotioning with successive refinement Voronoi lattice VQ for embedded wavelet image coding," in *Proc. ICIP*, 1998, vol. 1, pp. 107–111.

[19] C. Lamblin and J.-P. Adoul, "Algorithme de quantification vectorielle sphérique à partir du réseau de Gosset d'ordre 8," *Ann. Télécommun.*, vol. 43, no. 3–4, pp. 172–186, 1988.

[20] J.-M. Moureaux, P. Loyer, and M. Antonini, "Low-complexity indexing method for $\mathbb{Z}^n$ and D_n lattice quantizers," *IEEE Trans. Communications*, vol. 46, no. 12, Dec. 1998.

[21] P. Rault and C. Guillemot, "Lattice vector quantization with reduced or without look-up table," in *SPIE Electronic Imaging*, Santa Clara, FL, U.S.A., Jan. 1998.

[22] P. Rault and C. Guillemot, "Indexing algorithms for $\mathbb{Z}^n$, A_n, D_n, and D_n^{++} lattice vector quantizers," *IEEE Transactions on Multimedia*, 2001.

[23] F. Labonté, "Étude, optimisation et implémentation d'un quantificateur vectoriel algébrique encastré dans un codeur audio hybride ACELP/TCX," M.S. thesis, Université de Sherbrooke, Québec, Canada, Avril 2001.

[24] A. Gersho and R.M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic Publishers, 1992.

[25] R. Zamir, S.Shamai, and U.Erez, "Nested linear/lattice codes for structured multiterminal binning," *IEEE Trans. Inform. Theory*, vol. 48, no. 6, pp. 1250–1276, Jun. 2002.

Chapitre 5

CODAGE PAR TRONCATURE DE VORONOÏ ADAPTATIVE

Ce chapitre est dédié à une technique de quantification vectorielle algébrique multi-débit à faible complexité : le codage par troncature de Voronoï adaptative. Les dictionnaires de quantification utilisés sont imbriqués. Ces travaux ont été entrepris pendant le développement du modèle de codage AMR-WB+, avec pour objectif de développer une alternative à la quantification multi-débit par extension de Voronoï pour le codage algébrique de la cible TCX, qui est présentée au chapitre 4. La technique décrite ici est en fait moins complexe que son équivalent utilisant l'extension de Voronoï, tout en ayant des performances similaires.

La troncature de Voronoï adaptative est illustrée à la figure 5.1 pour le réseau D_2 en dimension 2. Cette figure montre les frontières de 6 régions de troncature emboîtées définissant des codes de Voronoï (des dictionnaires de quantification) de débits différents dans D_2.

On distingue deux types de régions :

- Les carrés ayant subi une rotation de $\pi/2$, qui correspondent à $2^n V(D_2) + \mathbf{a}$, où $V(D_2)$ est la région de Voronoï de D_2 et n est un entier ≥ 1 ; elles définissent un code de Voronoï dans D_2

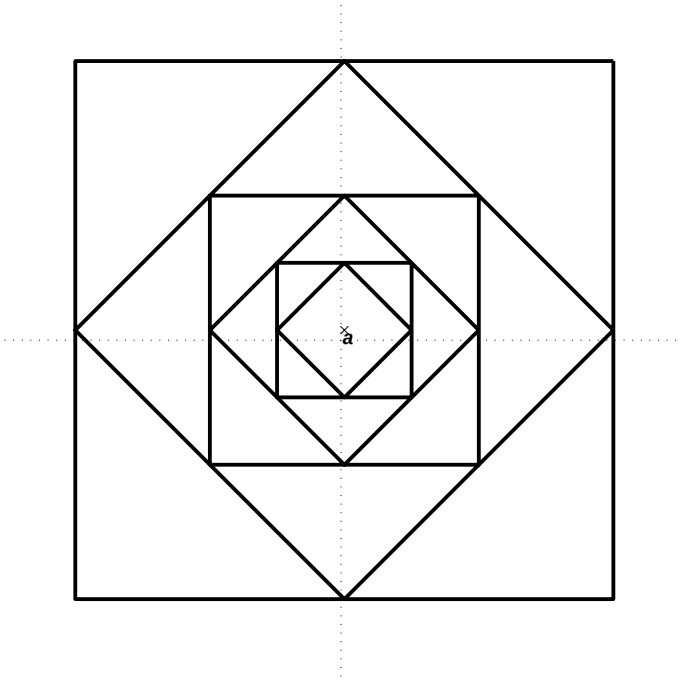


Figure 5.1: Régions de troncature de D_2.

de débit entier (n bit par dimension) ;

- Les autres carrés qui correspondent à $2^n V(2\mathbb{Z}^2) + \mathbf{a}$ et définissent un code de Voronoï dans D_2 de débit $n + \frac{1}{2}$ bit par dimension.

La figure 5.1 ne montre en fait que les cas $n = 1, 2$ et 3, avec $\mathbf{a} = (0.1, 0.3)$; cependant n peut aller en théorie jusqu'à l'infini. A partir de cet exemple, on comprend qu'un point quelconque $\mathbf{y}$ de D_2 peut être vu comme un élément d'un code de Voronoï dans D_2 de débit n ou $n + \frac{1}{2}$ bit par dimension, pourvu que n soit assez grand. Puisque n dépend de $\mathbf{y}$, la troncature de D_2 est dite adaptative. La troncature de Voronoï est mise ici à profit pour développer des algorithmes efficaces de quantification multi-débit – avec un stockage négligeable et un coût de calcul relativement faible et indépendant du débit de codage.

Ce chapitre est structuré comme suit. Les notions de base sur les partitions de réseaux de points

sont rappelées à la section 5.1. La troncature de Voronoï adaptative est ensuite présentée en 3 étapes. Elle est d'abord étudiée à la section 5.2 dans le cas scalaire, avec le réseau $\mathbb{Z}$. Le codage multi-débit est ainsi défini à l'aide de partitions de la forme $\mathbb{Z}/m\mathbb{Z}$ indexées sur $\log_2 m$ bits avec m entier ≥ 2. Ce cas particulier est ensuite généralisé à la section 5.3 pour un réseau régulier quelconque Λ en dimension N. Les partitions de la forme $\Lambda/m\Lambda$ sont indexées sur $\log_2 m$ bits par dimension. Des résultats numériques sont présentés à la section 5.4 afin d'évaluer les performances de codage. Enfin, le codage par incrément de $\frac{1}{2}$ bit par dimension est présenté à la section 5.5 à l'aide de partitions de la forme $\Lambda/2^r\Lambda/2^r R\Lambda$ où Λ est un réseau binaire en dimension paire, r est un entier ≥ 1 et $R\Lambda$ un sous-réseau géométriquement similaire à Λ tel que $|\Lambda/R\Lambda| = 2^{N/2}$. La technique décrite à la section 5.5 est appliquée à la section 5.6 pour développer un système de quantification multi-débit adapté aux besoins du codage AMR-WB+, avant de conclure à la section 5.7.

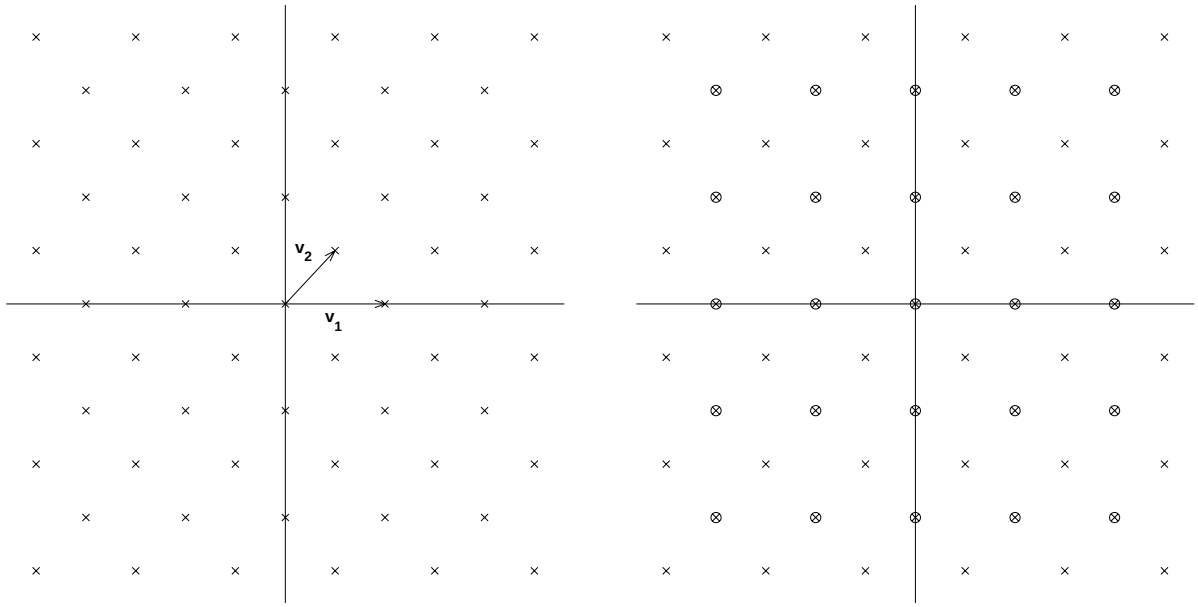
5.1 Préliminaires

Les notions qui sont revues ici ont été abordées dans les chapitres 2, 3 et 4. Un réseau de points Λ est un arrangement régulier de points dans l'espace, stable (clos) pour l'addition et la multiplication par des entiers [1]. On considère ici des réseaux définis dans $\mathbb{R}^N$ et de rang maximal, spécifiés par une base de vecteurs linéairement indépendants $\mathbf{v}_1 \cdots, \mathbf{v}_N$, avec

$$\Lambda = \left\{ k_1\mathbf{v}_1 + \cdots + k_N\mathbf{v}_N | \mathbf{k} = (k_1, \cdots, k_N) \in \mathbb{Z}^N \right\}. \tag{5.1}$$

Un exemple de réseau est montré (en partie) à la figure 5.2 (a). Il s'agit du réseau en damier en dimension $N = 2$, $D_2 = \mathbb{Z}\mathbf{v}_1 + \mathbb{Z}\mathbf{v}_2$, avec $\mathbf{v}_1 = (2, 0)$ et $\mathbf{v}_2 = (1, 1)$. Le réseau D_2 peut être également défini de façon plus intuitive comme $D_2 = \left\{ (x_1, x_2) \in \mathbb{Z}^2 | x_1 + x_2 \text{ pair} \right\}$.

Un sous-réseau Λ' de Λ est un réseau de points inclus dans Λ ($\Lambda' \subset \Lambda$). Par exemple, D_2 est un sous-réseau de $\mathbb{Z}^2$. De plus, $2\mathbb{Z}^2$ est un sous-réseau de D_2, tel que montré à la figure 5.2 (b). Un sous-réseau Λ' induit une partition de Λ en sous-ensembles dits *cosets*, qui sont des translatés de Λ'.



(a) réseau D_2 (b) coset $2\mathbb{Z}^2$ de D_2

Figure 5.2: Réseau D_2 et sous-réseau $2\mathbb{Z}^2$.

Ainsi, Λ peut alors être vu comme :

$$\Lambda = \bigcup_{\mathbf{c}\in[\Lambda/\Lambda']} \left(\Lambda' + \mathbf{c}\right), \tag{5.2}$$

où $[\Lambda/\Lambda']$ est un ensemble de vecteurs permettant de générer Λ par translation de Λ'. Ces vecteurs sont des *représentants de cosets*. Cette décomposition en cosets a été exploitée en particulier au chapitre 4. Par exemple, D_2 peut être partitionné sous la forme :

$$D_2 = 2\mathbb{Z}^2 \cup \left\{2\mathbb{Z}^2 + (1,1)\right\}. \tag{5.3}$$

Ainsi, on peut prendre $[D_2/2\mathbb{Z}^2] = \{(0,0), (1,1)\}$. La cardinalité de $[\Lambda/\Lambda']$ est notée $|\Lambda/\Lambda'|$ – autrement dit, $|\Lambda/\Lambda'|$ correspond au nombre de sous-ensembles dans la partition Λ/Λ'. Par exemple, $|D_2/2\mathbb{Z}^2| = 2$.

De façon évidente, le réseau $m\Lambda$ (m entier ≥ 2) est un sous-réseau de Λ. Plus précisément, il y

a m^N cosets de $m\Lambda$ dans Λ, soit $|\Lambda/m\Lambda| = m^N$. Cette propriété découle de la relation :

$$\Lambda = \bigcup_{\substack{0 \le k_i < m \\ i = 1, \cdots, N}} \left\{ m\Lambda + \underbrace{k_1\mathbf{v}_1 + \cdots + k_N\mathbf{v}_N}_{\text{représentant de coset de } m\Lambda} \right\}. \tag{5.4}$$

Par exemple, si $\Lambda = \mathbb{Z}$, on peut prendre $[\mathbb{Z}/m\mathbb{Z}] = \{0, 1, \cdots, m-1\}$. Le codage de Voronoï de [2] définit d'autres représentants de cosets pour la partition $\Lambda/m\Lambda$: les points de $[\Lambda/m\Lambda]$ sont pris à l'intérieur d'une région de Voronoï (décalée) de $m\Lambda$ en calculant $k_1\mathbf{v}_1 + \cdots + k_N\mathbf{v}_N$ modulo $m\Lambda$. On adopte les mêmes notations qu'au chapitre 4 ; un code de Voronoï $V(\Lambda, \Lambda')$ associé à la partition Λ/Λ' est défini par :

$$V(\Lambda, \Lambda') = \Lambda \cap \left\{ V(\Lambda') + \mathbf{a} \right\}, \tag{5.5}$$

où $\mathbf{a} \in \mathbb{R}^N$ est un décalage approprié. Si $\Lambda' = m\Lambda$, on a :

$$V(\Lambda, m\Lambda) = \Lambda \cap \left\{ mV(\Lambda) + \mathbf{a} \right\}. \tag{5.6}$$

Un réseau Λ dans $\mathbb{R}^N$ est dit binaire [3] si Λ est un sous-réseau de $\mathbb{Z}^N$ et s'il existe un entier $p \in \mathbb{N}$ tel que $2^p\mathbb{Z}^N$ est un sous-réseau de Λ. Le nombre de cosets de $2^p\mathbb{Z}^N$ de Λ est une puissance de 2. Pour N pair, un réseau binaire Λ admet comme sous-réseau $R\Lambda$, avec $|\Lambda/R\Lambda| = 2^{N/2}$, où R est une matrice de rotation adéquate vérifiant $\det R = 2^{N/2}$. Cette structure est détaillée dans [3], en particulier dans le cas des réseaux $\Lambda = D_N$, E_8, Λ_{16} et Λ_{24}.

5.2 Codage de Voronoï multi-débit dans $\mathbb{Z}/m\mathbb{Z}$

Le codage par troncature de Voronoï adaptative est illustré à la figure 5.3 dans le cas scalaire. Pour une source scalaire $x \in \mathbb{R}$ et en faisant abstraction de toute contrainte de débit, le codage de x dans $\mathbb{Z}$ s'effectue sans saturation (ou surcharge) : x est toujours représenté par son arrondi y dans $\mathbb{Z}$. Le point y est codé de façon adaptative par indexation multi-débit en s'appuyant sur le codage de

Voronoï scalaire. L'indexation produit un indice scalaire k ainsi qu'un numéro de dictionnaire m entier qui est ensuite codé sous forme binaire.

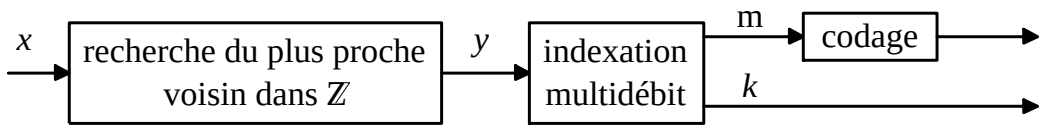


Figure 5.3: Codage multi-débit par troncature de Voronoï adaptative de $\mathbb{Z}$.

En dimension $N = 1$, le codage de Voronoï correspond en fait à la quantification scalaire uniforme, et un dictionnaire de taille fixe peut être défini comme

$$V(\mathbb{Z}, m\mathbb{Z}) = \mathbb{Z} \cap \{mV(\mathbb{Z}) + a\} = \mathbb{Z} \cap \left[-\frac{m}{2} + a, \frac{m}{2} + a\right], \tag{5.7}$$

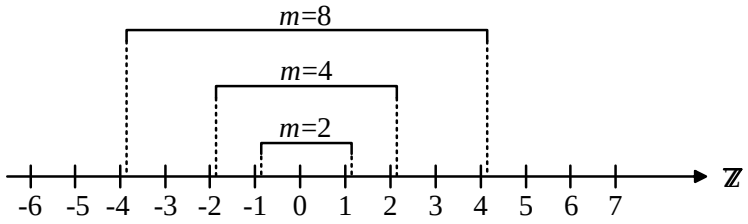
où $V(\mathbb{Z}) = \left[-\frac{1}{2}, \frac{1}{2}\right]$ est la région de Voronoi de $\mathbb{Z}$ associée à 0, m est un entier ≥ 2 et a est un décalage tel que $\pm\frac{m}{2} + a \notin \mathbb{Z}$. Des exemples de codes sont illustrés à la figure 5.4 (a) pour $a = 0.1$. Le code $V(\mathbb{Z}, m\mathbb{Z})$ contient m mots de code, qui sont des représentants de cosets de la partition $\mathbb{Z}/m\mathbb{Z}$. Le nombre de bits nécessaire pour indexer y dans $V(\mathbb{Z}, m\mathbb{Z})$ est $\lceil \log_2 m \rceil$, où $\lceil . \rceil$ désigne l'arrondi à l'entier supérieur. Par exemple, $V(\mathbb{Z}, 2\mathbb{Z}) = \{0, 1\}$, $V(\mathbb{Z}, 4\mathbb{Z}) = \{-1, 0, 1, 2\}$ et $V(\mathbb{Z}, 8\mathbb{Z}) = \{-3, -2, -1, 0, 1, 2, 3, 4\}$; ces codes (ou dictionnaires) comprennent respectivement 2, 4 et 8 mots de code, pour un débit respectif de 1, 2 et 3 bits. On vérifie que les dictionnaires sont imbriqués, car $V(\mathbb{Z}, 2\mathbb{Z}) \subset V(\mathbb{Z}, 4\mathbb{Z}) \subset V(\mathbb{Z}, 8\mathbb{Z})$.

5.2.1 Algorithme de codage

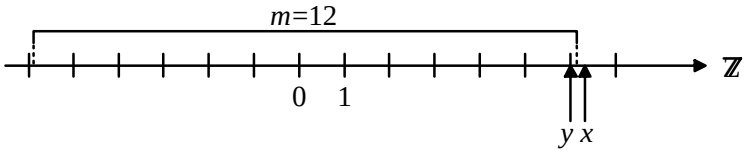
Le codage multi-débit d'un scalaire x dans $V(\mathbb{Z}, m\mathbb{Z})$, schématisé à la figure 5.3, peut être effectué en trois étapes :

1. Arrondi de x à l'entier y le plus proche, soit calcul de $y = [x]$ où $[.]$ désigne la partie entière ;

2. Recherche de la troncature minimale de $\mathbb{Z}$ pour que y soit un mot de $V(\mathbb{Z}, m\mathbb{Z})$, soit recherche



(a) exemples de code de Voronoï $V(\mathbb{Z}, m\mathbb{Z})$ pour $m = 2, 4, 8$



(b) exemple de codage

Figure 5.4: Exemples de codes issus de la partition $\mathbb{Z}/m\mathbb{Z}$.

de

$$m_{min} = \min\left\{m \in \mathbb{N}\backslash\{0,1\} \mid -\frac{m}{2} + a < y < \frac{m}{2} + a\right\} \tag{5.8}$$

– on trouve facilement que $|m| > 2|y - a|$ et donc $m_{min} = max(\lceil 2|y-a| \rceil, 2)$;

3. Calcul de l'indice $0 \leq k < m$ de y dans $V(\mathbb{Z}, m\mathbb{Z})$ pour $m = m_{min}$ (par exemple $k = y - \lceil -\frac{m_{min}}{2} + a \rceil$).

Un exemple de codage est illustré à la figure 5.4 (b) pour $a = 0.1$ et $x = 6.31$. On trouve $y = 6$, $m_{min} = 12$ et $k = 11$.

Le modulo m est une information supplémentaire, nécessaire pour décoder correctement k. En pratique, pour limiter le débit supplémentaire dû au codage de m, la recherche de m_{min} peut être contrainte sur un sous-ensemble de $\mathbb{N}\backslash\{0, 1 \cdots\}$. Par exemple, on peut limiter m à des puissances de 2, c'est-à-dire à $\{2, 4, 8, 16, \cdots\}$ pour obtenir des incréments de débit de 1 bit. Par ailleurs, la contrainte sur a, à savoir que a soit tel que $\pm\frac{m}{2} + a \notin \mathbb{Z}$, n'est que théorique. En pratique, on peut très bien prendre $a = 0$; la recherche de m_{min} et le calcul de k doivent être cependant modifiés pour

s'assurer du bon fonctionnement du codage.

5.2.2 Variante : dictionnaires emboîtés

Les constellations de Voronoï $V(\mathbb{Z}, m\mathbb{Z}) = \mathbb{Z} \cap \{mV(\mathbb{Z}) + a\}$ ont deux défauts :

- Elles sont asymétiques et décentrées (de moyenne non nulle) – cette propriété est inévitable si on impose que 0 soit un mode de code ;
- Elles sont imbriquées ($V(\mathbb{Z}, 2\mathbb{Z}) \subset V(\mathbb{Z}, 3\mathbb{Z}) \subset V(\mathbb{Z}, 4\mathbb{Z}) \subset \cdots$), l'indexation multi-débit est donc redondante.

Pour remédier à ce dernier défaut, on peut utiliser des constellations emboîtées. Par exemple, si la valeur de $m \geq 2$ est contrainte à une puissance de 2, on peut définir C_m tel que $C_m = \mathbb{Z} \cap V_m$, avec $V_2 = 2V(\mathbb{Z}) + a$ et $V_m = \left\{\frac{m}{2}V(\mathbb{Z}) + (a - \frac{m}{2})\right\} \cup \left\{\frac{m}{2}V(\mathbb{Z}) + (a + \frac{m}{2})\right\}$ pour $m > 2$. Si $a = 0.1$, on obtient $C_2 = \{0, 1\}$, $C_4 = \{-2, -1, 2, 3\}$, $C_8 = \{-6, -5, -4, -3, 4, 5, 6, 7\}$, etc. Ces codes alternatifs sont dits emboîtés car $C_{m_1} \cap C_{m_2} = \emptyset$ si $m_1 \neq m_2$. Le codage est alors légèrement plus compliqué et il se généralise difficilement à d'autres réseaux que $\mathbb{Z}$.

5.3 Codage de Voronoï multi-débit dans $\Lambda/m\Lambda$

On généralise ici le codage de Voronoï multi-débit tel qu'introduit précédemment pour $\mathbb{Z}$ au cas d'un réseau régulier de points Λ quelconque en dimension N. Tel qu'illustré à la figure 5.5, le vecteur de source $\mathbf{x} \in \mathbb{R}^N$ est représenté par son plus proche voisin $\mathbf{y}$ dans Λ, et la performance de codage est mesurée par $\|\mathbf{x} - \mathbf{y}\|^2$ en moyenne (en supposant un budget de bits suffisant).

L'indexation multi-débit consiste à interpréter $\mathbf{y}$ comme un mot dans un code de Voronoï $V(\Lambda, m\Lambda)$ dans Λ avec

$$V(\Lambda, m\Lambda) = \Lambda \cap \{mV(\Lambda) + \mathbf{a}\}, \tag{5.9}$$

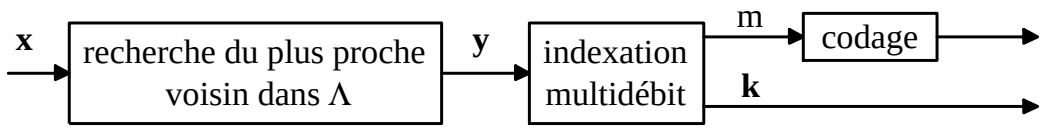


Figure 5.5: Codage multi-débit par troncature de Voronoï adaptative de Λ.

où m est un entier ≥ 2 adéquat et $\mathbf{a} \in \mathbb{R}^N$ est un décalage indépendant de m. Le code $V(\Lambda, m\Lambda)$ contient en fait des représentants de cosets de la partition $\Lambda/m\Lambda$ [4, 5]. L'indice $\mathbf{k}$ de $\mathbf{y}$ est vectoriel, et ses composantes vérifient $0 \leq k_i < m$ pour $i = 1, \cdots, N$. Cet indice a été défini aux chapitres 2, 3 et 4. Le modulo de Voronoï m est une information supplémentaire nécessaire au décodage de $\mathbf{k}$. Le nombre de bits utilisé pour représenter $\mathbf{k}$ est de $\log_2 m$ bits par dimension – en pratique $\lceil N \log_2 m \rceil$ bits par vecteur.

Ce codage multi-débit fonctionne correctement à condition que $\mathbf{y} \in V(\Lambda, m\Lambda)$. Idéalement on cherche

$$m_{min} = \min \{ m \in \mathbb{N} \backslash \{0, 1\} \, | \mathbf{y} \in (mV(\Lambda) + \mathbf{a}) \}, \tag{5.10}$$

pour utiliser le moins de bits possible au codage.

5.3.1 Algorithme de codage optimisé

On suppose pour simplifier que le décalage $\mathbf{a}$ est tel qu'aucun point de Λ n'est sur la frontière de $mV(\Lambda) + \mathbf{a}$ et que $\mathbf{a} \in V(\Lambda)$. La valeur $m = 0$ est permise pour traiter les cas où le budget de bits n'est pas suffisant au codage. Les étapes de codage sont alors :

1. Recherche du plus proche voisin $\mathbf{y}$ de $\mathbf{x}$ dans Λ ;

2. Recherche de la troncature minimale m_{min} :

   Si $\mathbf{y} = (0, \cdots, 0)$, $m_{min} = 0$, sinon :

(a) Recherche du vecteur pertinent $\mathbf{r}$ de Λ associé à $\mathbf{y}-\mathbf{a}$ (on peut se référer au chapitre 3 pour des détails sur cette recherche du vecteur pertinent) ;

(b) Calcul de $m_{min} = max(2, \lceil \frac{2(\mathbf{y}-\mathbf{a})\mathbf{r}^t}{\mathbf{r}\mathbf{r}^t} \rceil)$;

3. Calcul de l'indice $\mathbf{k}$ de $\mathbf{y}$ dans $V(\Lambda, m_{min}\Lambda)$.

Pour $\Lambda = \mathbb{Z}^N$ ($N \geq 1$), il y a $2N$ vecteurs pertinents possibles qui correspondent aux permutations de $(\pm 2, 0, \cdots, 0)$. La recherche du vecteur pertinent associé à $\mathbf{y}-\mathbf{a}$ revient ainsi à trouver la composante maximale en valeur absolue, et $m_{min} = 2|y_j - a_j|$ avec $j = \arg \max_{1 \leq i \leq N} |y_i - a_i|$.

Le calcul de m_{min} peut être compris à l'aide de la figure 5.6. En effet, pour que $\mathbf{y}$ soit dans $mV(\Lambda)+\mathbf{a}$, m doit être tel que $\|\mathbf{y}-\mathbf{a}\|^2 < \|\mathbf{y}-(m\mathbf{r}+\mathbf{a})\|^2$. On trouve : $(\mathbf{y}-\mathbf{a}-\frac{m}{2}\mathbf{r})\mathbf{r}^t < 0$. Cette condition est évidente d'un point de vue géométrique, comme illustré à la figure 5.6.

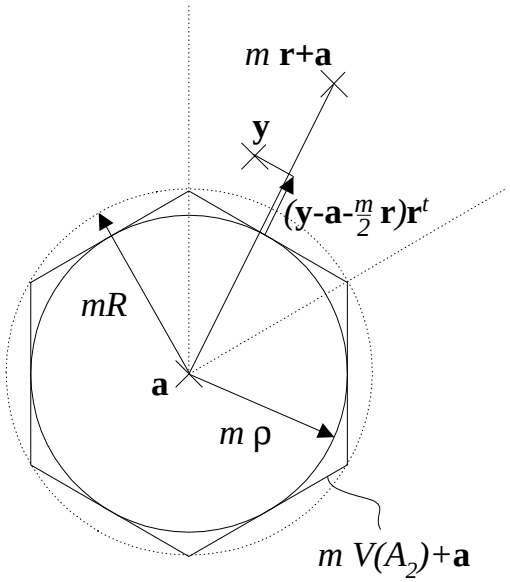


Figure 5.6: Détermination de la troncature minimale – exemple de A_2.

Pour réduire la complexité de codage, l'étape 2 peut être remplacée par une estimation sous-optimale de m_{min}. En effet, en remplaçant $mV(\Lambda)+\mathbf{a}$ par deux sphères, l'une inscrite et l'autre circonscrite, centrées en $\mathbf{a}$ de rayon $m\rho$ et mR respectivement, on trouve $m\rho < \|\mathbf{y}-\mathbf{a}\| < mR$ où ρ

est le rayon d'empilement de Λ et R est le rayon de couverture de Λ. Par suite, m_{min} est compris entre $\lfloor \frac{1}{\rho}\|\mathbf{y}-\mathbf{a}\| \rfloor$ et $\lceil \frac{1}{\rho}\|\mathbf{y}-\mathbf{a}\| \rceil$.

Ce codage est illustré à la figure 5.7 pour $\Lambda = A_2$: en prenant $\mathbf{x} = (1.2, 5.9)$ et $\mathbf{a} = (0.1, 0)$, on trouve $\mathbf{y} = (\frac{3}{2}, \frac{7\sqrt{3}}{2})$, $\mathbf{r} = (\frac{1}{2}, \frac{\sqrt{3}}{2})$ et $m_{min} = 12$; l'indice $\mathbf{k}$ peut être calculé par indexation de Voronoï [4]. En approximant de $mV(A_2) + \mathbf{a}$ par deux sphères centrée en $\mathbf{a}$ et de rayon $m\rho$ et mR, avec $\rho = \frac{1}{2}$ et $R = \frac{1}{\sqrt{3}}$, on trouve que m_{min} est compris entre 10 et 13 (pour une valeur réelle de 12).

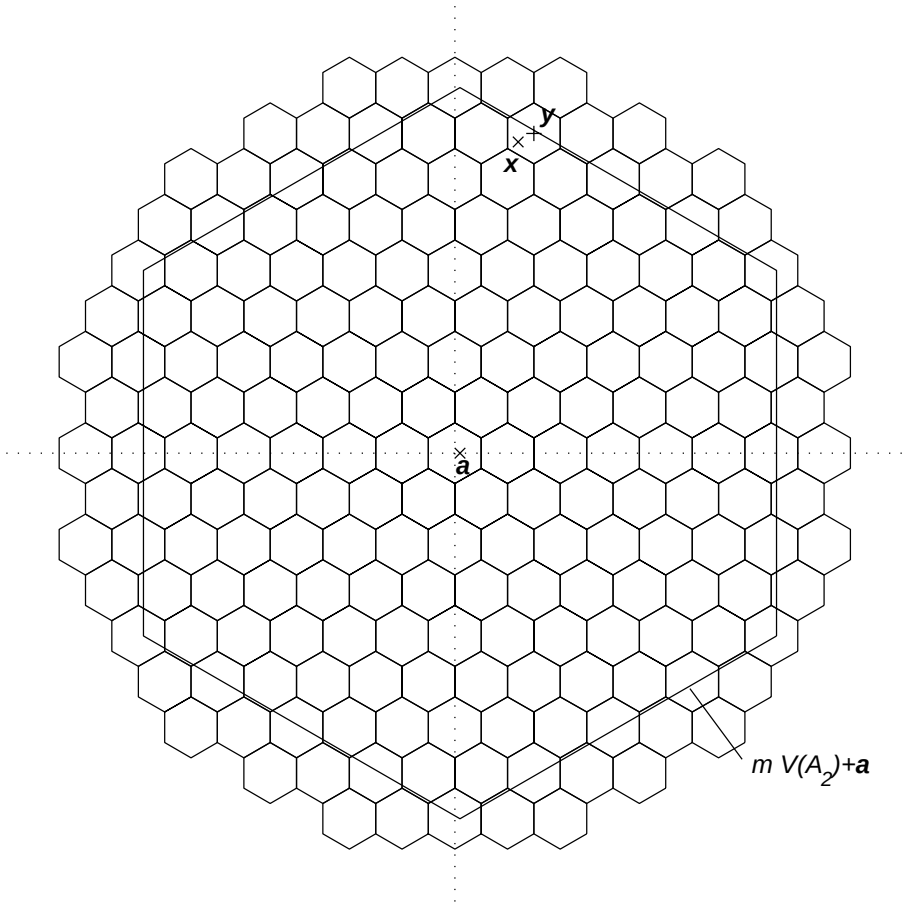


Figure 5.7: Exemple de troncature adaptative pour A_2/mA_2.

Le facteur m peut être restreint à un sous-ensemble de $\mathbb{N}\setminus\{0,1\}$. Le décalage $\mathbf{a}$ peut être quelconque à condition de définir correctement le calcul de m_{min} et de l'indice $\mathbf{k}$.

5.3.2 Algorithme de codage simplifié

La recherche du vecteur pertinent de Λ dans l'algorithme de codage optimisé peut être évitée en employant un algorithme simplifié décrit ci-dessous :

1. Recherche du plus proche voisin $\mathbf{y}$ de $\mathbf{x}$ dans Λ ;
2. Recherche de la troncature minimale m_{min} :

   Si $\mathbf{y} = (0, \cdots, 0)$, $m_{min} = 0$, sinon :

   (a) $m = 2$

   (b) recherche du plus proche voisin $\mathbf{z}$ de $(\mathbf{y} - \mathbf{a})/m$ dans Λ

   (c) si $\mathbf{z} = (0, \cdots, 0)$, $m_{min} = m$. Fin.

   (d) $m := m + 1$ et retour à l'étape (b)
3. Calcul de l'indice $\mathbf{k}$ de $\mathbf{y}$ dans $V(\Lambda, m_{min}\Lambda)$.

Les étapes (b) et (c) peuvent être remplacées à l'aide de l'indexation de Voronoï en calculant $\mathbf{y} \rightarrow \mathbf{k} \rightarrow \mathbf{z}$, où $\mathbf{k}$ est l'indice de Voronoï de $\mathbf{y}$, et en vérifiant que $\mathbf{y}$ et $\mathbf{z}$ sont identiques. Ces opérations ont été utilisées aux chapitres 3 et 4.

Avec cet algorithme simplifié, le nombre de boucles sur m n'est alors pas borné et la complexité de codage dépend de $\mathbf{y}$, donc de $\mathbf{x}$. Par contre, contrairement à l'algorithme optimisé, le décalage $\mathbf{a}$ peut prendre n'importe quelle valeur, à condition de remplacer les étapes (b) et (c) à l'aide des opérations $\mathbf{y} \rightarrow \mathbf{k}$ et $\mathbf{k} \rightarrow \mathbf{y}$.

5.3.3 Complexité

Le codage de Voronoï, tel que décrit dans [4], présente l'avantage d'avoir une complexité indépendante du débit de codage. La quantification multi-débit à partir de $\Lambda/m\Lambda$ conserve cette propriété, si

bien qu'avec l'algorithme de codage optimisé, la complexité de codage et décodage est indépendante de m, donc du débit. De plus, en mettant de côté le codage de m, les données à stocker pour réaliser la quantification dans $\Lambda/m\Lambda$ se réduisent essentiellement à $M(\Lambda)$, $M(\Lambda)^{-1}$ et $\mathbf{a}$. Cette complexité réduite est comparable à celle du codage de Voronoï quasi-ellipsoïdal vu au chapitre 3.

5.3.4 Remarque : estimation approximative de la troncature minimale

On décrit ici une technique due à Ohm, présentée dans [6] et similaire au codage par troncature de Voronoï adaptative. Celle-ci a été mise en œuvre dans [6] pour le codage vidéo.

La différence entre la quantification multi-débit de [6] et le codage de Voronoï à troncature adaptative développé ici pour $\Lambda/m\Lambda$ tient à la façon dont la troncature de Voronoï est adaptée. Plus précisément, dans [6], le calcul du modulo de Voronoï minimal m_{min} est approximatif.

L'estimation de m_{min} est appelée "adaptation de la taille du dictionnaire" dans [6] ; elle s'appuie sur des arguments géométriques. Le code de Voronoï $V(\Lambda, m\Lambda)$ est vu comme un code sphérique de rayon r ; la valeur de r est calculée pour un $\mathbf{x}$ donné et sert à trouver m, donc à adapter la troncature de Voronoï. L'estimation de m à partir de $\mathbf{x}$ n'étant pas détaillée dans [6], on reprend ici les arguments de [6] en développant les calculs.

Pour un $\mathbf{x}$ donné et en supposant que le décalage de Voronoï $\mathbf{a}$ vérifie $\|\mathbf{a}\|^2 \approx 0$, un code "sphérique" $V(\Lambda, m\Lambda)$ "englobant" $\mathbf{x}$ doit avoir un rayon r tel que $r \geq \|\mathbf{x}\|$, où $\|.\|$ désigne la norme euclidienne. Puisqu'une (hyper-)sphère en dimension N de rayon r a pour volume

$$V = \frac{\pi^{N/2} r^N}{\Gamma(N/2+1)}, \tag{5.11}$$

et que le volume d'une région de Voronoï $V(\Lambda)$ du réseau Λ est $detM(\Lambda)$ où $M(\Lambda)$ est une matrice génératrice de Λ, le nombre de mots de code dans $V(\Lambda, m\Lambda)$ pour représenter $\mathbf{x}$ est approximativement $V/det(M(\Lambda))$. Or, la taille du code de Voronoï $V(\Lambda, m\Lambda)$ est m^N [4] ; le modulo de Voronoï m minimal peut donc être estimé par $m_{est} = \sqrt[N]{V/det(M(\Lambda))}$ en prenant $r = \|\mathbf{x}\|$. Dans l'exemple précédent basé sur A_2, pour $\mathbf{x} = (1.2, 5.9)$ on trouve $m_{est} = 11.46$.

Dans [6], $\Lambda = \Lambda_{16}$ est utilisé pour des débits inférieurs à 4 bits par dimension ; autrement le codage est effectué dans $\Lambda = E_8$. Le facteur m prend ses valeurs dans $\{2, 3, 4, 6, 8, 12, \cdots\}$ pour que la taille du code $V(\Lambda, m\Lambda)$ augmente avec des incréments proches de $\frac{1}{2}$ bits par dimension, et m est représenté par un code de Huffman. Le décalage $\mathbf{a}$ définissant $V(\Lambda, m\Lambda)$ n'est pas précisé – la valeur maximale de m non plus.

5.4 Performances de quantification pour une source gaussienne

Les performances du codage par troncature de Voronoï adaptative sont évaluées pour une source discrète gaussienne sans mémoire, de moyenne nulle et de variance unité. Les codes définis par la troncature de Voronoï adaptative sont en effet quasi-sphériques, à condition d'utiliser un réseau tel que A_2, D_4, RE_8, Λ_{16} ou Λ_{24}. L'importance d'adapter la forme des dictionnaires de quantification à la source a été discutée au chapitre 2. En particulier, pour une source gaussienne sans mémoire, les dictionnaires doivent être sphériques. Or, la forme de région de $V(\Lambda)$ s'approche de celle d'une sphère ; la troncature quasi-sphérique utilisée ici est donc appropriée. En outre, la source gaussienne fournit une approximation réaliste pour certains signaux tels que la cible TCX transformée [7] (pour un codage TCX avec prédiction de pitch).

On se restreint au cas du codage multi-débit à débits entiers à partir de $\Lambda/2^n\Lambda$. Seuls les réseaux $\Lambda = \mathbb{Z}$ et A_2 sont utilisés. Les simulations utilisent dans les deux cas 10^6 vecteurs de source.

La source est codée selon un schéma de type gain-forme, tel qu'illustré à la figure 5.8. Les dictionnaires de quantification multi-débit sont notés Q_n, avec $n \geq 0$, comme à la section 4.4. Ils sont définis ici par : $Q_0 = \{(0, \cdots, 0)\}$ et $Q_n = V(\Lambda, 2^n\Lambda)$ (pour $n > 0$) en prenant comme paramètre de troncature adaptative $m = 2^n$. Le débit de Q_n est ainsi égal à n bits par dimension. L'information supplémentaire $n = \log_2 m$ est représentée par un code binaire à longueur variable ; on emploie un codage entropique idéal ou un codage unaire. On suppose que le budget de bits n'est pas limité pour coder chaque vecteur de source ; par suite, la distorsion est réduite à la distorsion granulaire $\|\mathbf{x} - g\mathbf{y}\|^2$, où $\mathbf{y}$ est le plus proche voisin de $\frac{1}{g}\mathbf{x}$ dans Λ. Le point $\mathbf{y}$ est codé sous la forme

d'un numéro de dictionnaire n et d'un indice i identifiant un mot de code de Q_n. L'indice i est formé en multiplexant les composantes de l'indice de Voronoï $\mathbf{k}$ associé à $Q_n = V(\Lambda, 2^n\Lambda)$ pour $n > 0$.

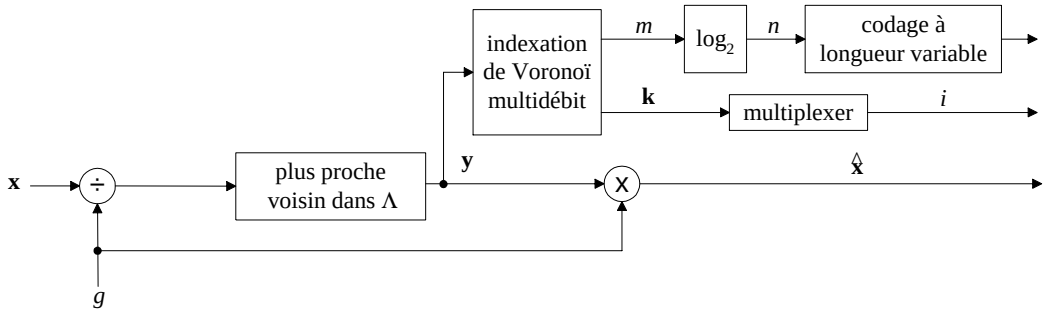


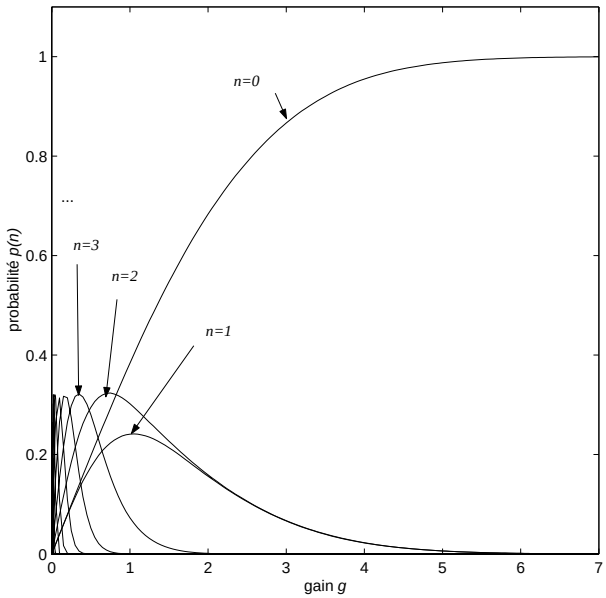
Figure 5.8: Système d'évalutation.

En faisant varier g, on mesure le débit moyen de codage R et la distorsion moyenne normalisée $D = \frac{1}{N}E\left[\|\mathbf{x} - g\mathbf{y}\|^2\right]$. On obtient ainsi une courbe débit-distorsion $D(R)$. Pour chaque valeur de g, on estime également la probabilité de sélection des dictionnaires Q_n, ainsi que la longueur moyenne $E[l(n)]$ du code à longueur variable représentant n.

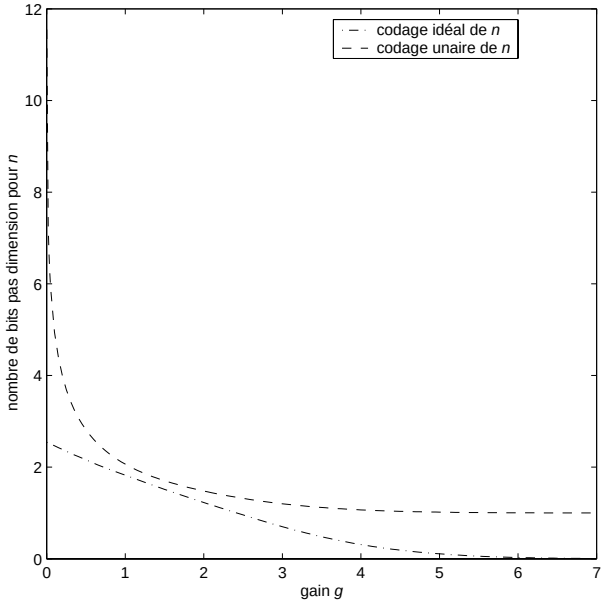
5.4.1 Quantification multi-débit dans $\mathbb{Z}$

Les performances mesurées pour $\mathbb{Z}$ sont illustrées à la figure 5.9. On vérifie à la figure 5.9 (a), que pour un gain g assez élevé, la source est majoritairement localisée dans $V(\Lambda)$ et donc codée à $(0, \cdots, 0)$, avec $n = 0$. La probabilité de sélection de chaque dictionnaire Q_n peut être évaluée analytiquement à l'aide de la fonction d'erreur (erf) en fonction de g. La sous-optimalité du codage unaire (par rapport à un codage entropique idéal) est mise en évidence à la figure 5.9 (b). Avec le codage unaire, au minimum un bit doit être transmis pour chaque $\mathbf{x}$ pour indiquer le dictionnaire Q_n utilisé. Le débit moyen par dimension R tend donc vers 1 bit pour les grandes valeurs de g.

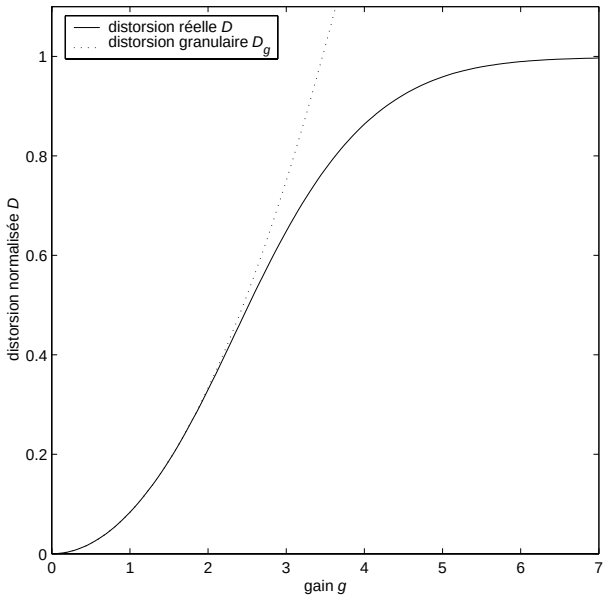
Un gain g faible implique un débit moyen élevé. A la figure 5.9 (c), on voit que la distorsion se réduit à la distorsion granulaire à débit suffisamment élevé (pour g suffisamment faible). La distorsion est alors approximativement égale au moment d'ordre 2 de $g\mathbb{Z}$, soit $D \approx D_g = g^2/12$. La



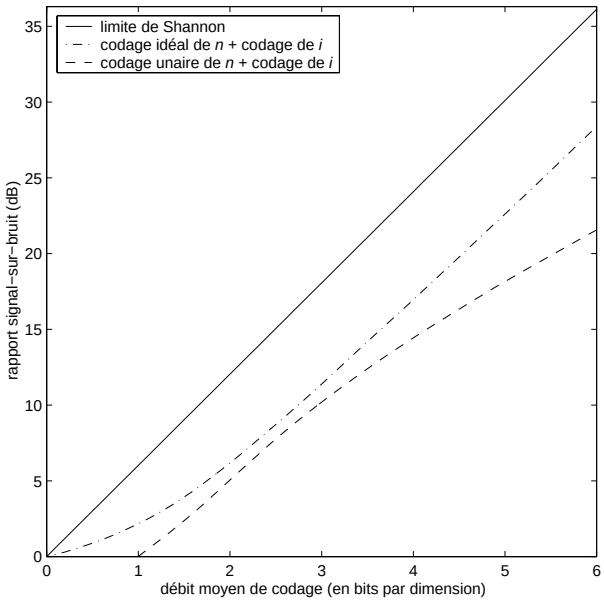
(a) probabilité de sélection



(b) $E[l(n)]$



(c) distorsion



(d) rapport signal-sur-bruit

Figure 5.9: Résultats de simulation pour la quantification multi-débit dans Z.

figure 5.9 (d) montre que les performances du codage dans $\mathbb{Z}$ s'écartent significativement de la borne débit distorsion quand le débit augmente. Cet écart peut être attribué à plusieurs raisons :

- Pour chaque $\mathbf{x}$ scalaire, il faut coder une information supplémentaire n. Quand g diminue, $E[l(n)]$ augmente ; le débit relatif consommé par le codage de n pénalise la performance.
- Les dictionnaires Q_n sont imbriqués.
- Le codage est effectué dans $\mathbb{Z}$ qui n'apporte aucun gain granulaire.

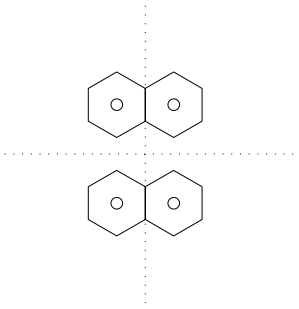
5.4.2 Quantification multi-débit dans A_2

Le codage dans A_2 est effectué avec l'algorithme optimisé d'indexation multi-débit et en prenant $\mathbf{a} = (0.1, 0)$. Les performances associées sont montrées à la figure 5.11. On peut vérifier qu'elles sont équivalentes à celles obtenues avec la quantification multi-débit par extension de Voronoï (qui sont analysées au paragraphe 4.4.1).

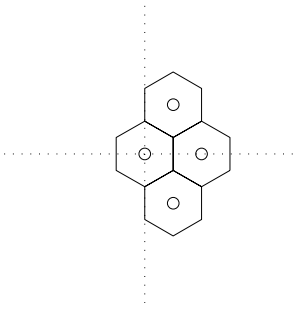
Néanmoins à bas débit (autour d'un bit par dimension) les performances sont moins bonnes que celles obtenues avec l'extension de Voronoï ; cette dégradation peut être expliquée par l'asymétrie et la moyenne non nulle des dictionnaires Q_1 et Q_2, comme illustrés à la figure 5.10.

5.5 Codage de Voronoï multi-débit dans $\Lambda/2^r\Lambda/2^r R\Lambda$ où Λ est un réseau binaire

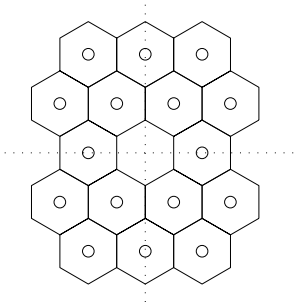
Le codage à partir de $\Lambda/m\Lambda$, décrit à la section 5.3, peut être adapté pour que le débit des codes de Voronoï augmente avec un incrément de $\frac{1}{2}$ bit par dimension. On complète ici les algorithmes de codage dans $\Lambda/m\Lambda$ afin de mettre en œuvre un système de quantification multi-débit, adapté au codage TCX audio, présenté à la section 5.6.



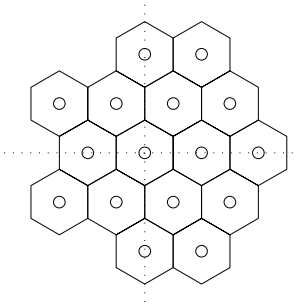
(a) Q_1 pour l'extension de Voronoï



(b) Q_1 pour le codage par troncature de Voronoï adaptative



(a) Q_2 pour l'extension de Voronoï

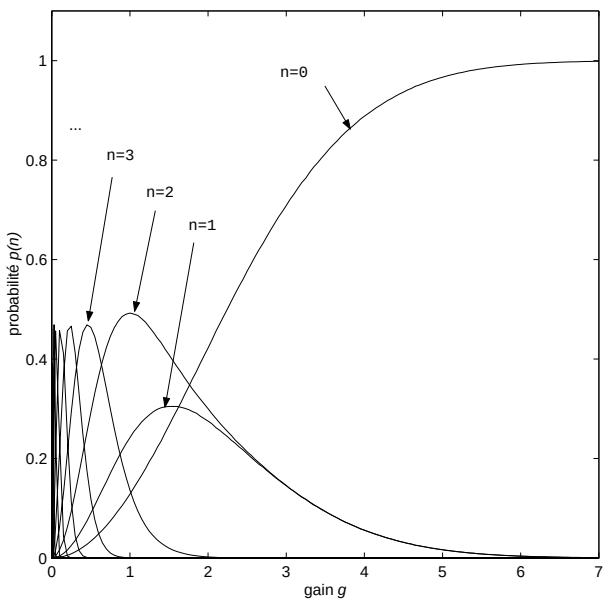


(b) Q_2 pour le codage par troncature de Voronoï adaptative

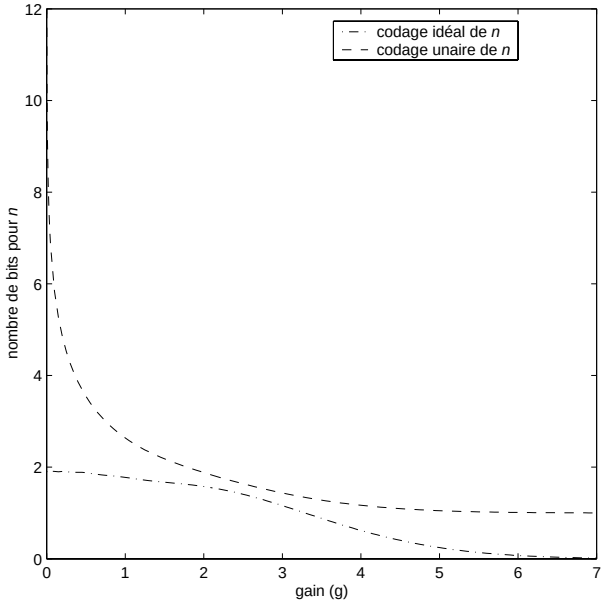
Figure 5.10: Comparaison des dictionnaires Q_1 et Q_2.

Au lieu d'utiliser des partition $\Lambda/2^r\Lambda$ (avec r entier ≥ 1) conduisant à des incréments de 1 bit par dimension, on utilise les partition $\Lambda/2^r\Lambda/2^r R\Lambda$ avec $|\Lambda/R\Lambda| = 2^{N/2}$ et $R^2\Lambda = 2\Lambda$. Le réseau Λ doit dans ce cas être binaire en dimension paire. Les réseaux D_N (N entier pair), RE_8, Λ_{16} et Λ_{24} sont des exemples de réseaux applicables.

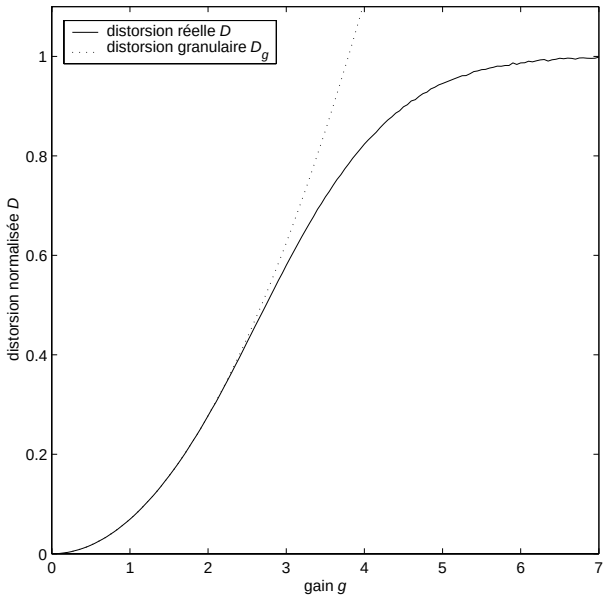
Le codage dans $\Lambda/2^r\Lambda/2^r R\Lambda$ est similaire au codage dans $\Lambda/m\Lambda$ pour $m = 2^r$. Il est illustré à la figure 5.12. Le vecteur de source $\mathbf{x} \in \mathbb{R}^N$ est représenté par son plus proche voisin $\mathbf{y}$ dans Λ, et la performance de codage est mesurée en moyennant $\|\mathbf{x} - \mathbf{y}\|^2$ (en supposant un budget de bits suffisant). Le point $\mathbf{y}$ est indexé dans des dictionnaires Q_n sous la forme d'un numéro de dictionnaire n et d'un indice i qui est obtenu en multiplexant un indice de Voronoï $\mathbf{k}$ ou des indices vectoriels $\mathbf{k}$



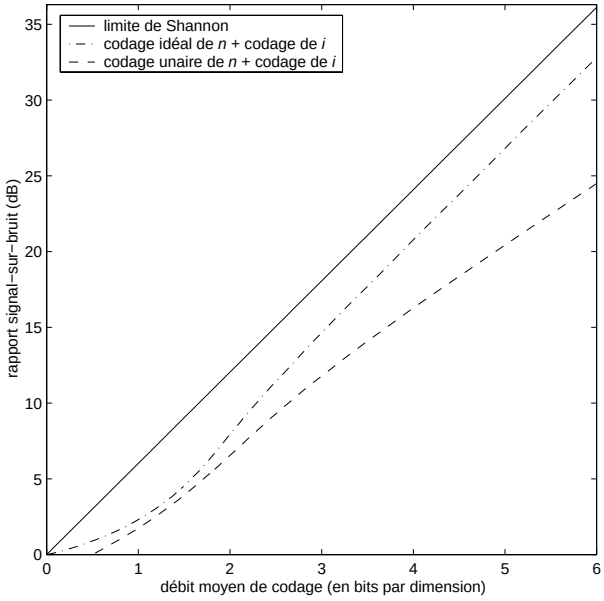
(a) probabilité de sélection



(b) $E[l(n)]$



(c) distorsion



(d) rapport signal-sur-bruit

Figure 5.11: Résultats de simulation pour la quantification multi-débit dans A_2.

et **l**.

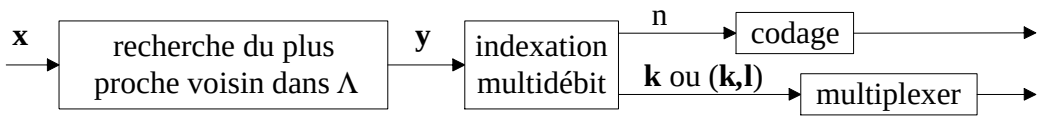


Figure 5.12: Codage multi-débit par troncature de Voronoï adaptative de Λ.

Les dictionnaires sont des codes de Voronoï dans Λ, définis pour r entier ≥ 1 par :

$$Q_{2r} = V(\Lambda, 2^r\Lambda) = \Lambda \cap \{V(2^r\Lambda) + \mathbf{a}\} = \Lambda \cap \{2^r V(\Lambda) + \mathbf{a}\}, \tag{5.12}$$

$$Q_{2r+1} = V(\Lambda, 2^r R\Lambda) = \Lambda \cap \{V(2^r R\Lambda) + \mathbf{a}\} = \Lambda \cap \{2^r V(R\Lambda) + \mathbf{a}\}, \tag{5.13}$$

où $\mathbf{a} \in \mathbb{R}^N$ est un décalage indépendant de r. Ces deux équations peuvent être résumées, pour n entier ≥ 2, sous la forme :

$$Q_n = V(\Lambda, R^n\Lambda) = \Lambda \cap \{V(R^n\Lambda) + \mathbf{a}\}, \tag{5.14}$$

en se rappelant que $R^{2n}\Lambda = 2^n\Lambda$ et $R^{2n+1}\Lambda = 2^n R\Lambda$. Chaque dictionnaire Q_n comprend $2^{\frac{n}{2}N}$ éléments, pour un débit de $\frac{n}{2}$ bits par dimension. Par convention on fixe le débit minimal à 1 bit par dimension ; mais en réalité la valeur $n = 1$ pourrait aussi être utilisée. On définit également le dictionnaire $Q_0 = \{0, \cdots, 0\}$.

5.5.1 Algorithme de codage optimisé

On suppose que le décalage $\mathbf{a}$ est tel qu'aucun point de Λ n'est sur la frontière de $2^r V(\Lambda) + \mathbf{a}$ ou $2^r V(R\Lambda) + \mathbf{a}$ et que $\mathbf{a} \in V(\Lambda)$. Le codage dans $\Lambda/m\Lambda$, décrit au paragraphe 5.3.1, est généralisé au cas d'une partition $\Lambda/2^r\Lambda/2^r R\Lambda$ ci-dessous :

1. Recherche du plus proche voisin $\mathbf{y}$ de $\mathbf{x}$ dans Λ.

2. Si $\mathbf{y} = (0, \cdots, 0)$, $n = 0$. Fin.

3. Trouver r_1 minimal tel que $\mathbf{y} \in V(\Lambda, 2^{r_1}\Lambda)$ à l'aide de la recherche du vecteur pertinent $\mathbf{r}$ de Λ associé à $\mathbf{y} - \mathbf{a}$.

4. Trouver r_2 minimal tel que $\mathbf{y} \in V(\Lambda, 2^{r_2}R\Lambda)$ à l'aide de la recherche du vecteur pertinent $\mathbf{r}$ de $R\Lambda$ associé à $\mathbf{y} - \mathbf{a}$.

5. Sélectionner la troncature minimale : si $r_2 < r_1$, $n = 2r_2 + 1$, sinon $n = 2r_1$.

6. Calculer l'indice de $\mathbf{y}$ dans Q_n : si n est pair, ce calcul correspond à l'indexation de Voronoï de [4]. Si n est impair, l'indexation peut être tirée de [5].

Cet algorithme est illustré à la figure 5.13. La quantification s'effectue dans D_2 avec

$$R = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}. \tag{5.15}$$

On rappelle que $RD_2 = 2\mathbb{Z}^2$. On prend $\mathbf{a} = (0.3, 0.1)$. Pour $\mathbf{x_1} = (2.4, 3.6)$, on trouve $\mathbf{y_1} = (2, 4)$, $n = 5$. Pour $\mathbf{x_1} = (-1.9, 5.6)$, on trouve $\mathbf{y_2} = (-2, 6)$, $n = 6$.

5.5.2 Algorithme de codage simplifié

La recherche du vecteur pertinent de Λ ou $R\Lambda$ peut être évitée avec l'algorithme ci-dessous :

1. Recherche du plus proche voisin $\mathbf{y}$ de $\mathbf{x}$ dans Λ ;

2. Si $\mathbf{y}$ est nul, $n = 0$. Fin.

3. Rechercher la troncature minimale m_{min} :

   (a) $n = 2$, $m = 2$

   (b) Si $\mathbf{y} \in Q_n = V(\Lambda, m\Lambda)$, arrêter ;

   (c) $n := n + 1$

   (d) Si $\mathbf{y} \in Q_n = V(\Lambda, mR\Lambda)$, arrêter ;

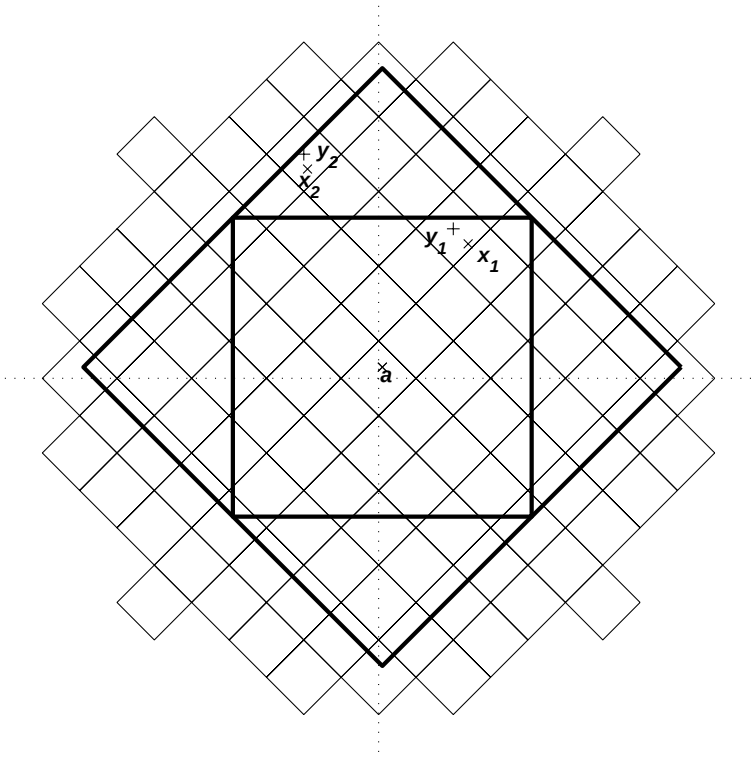


Figure 5.13: Exemple de troncature adaptative pour D_2/R^pD_2.

(e) $n := n + 1$, $m := 2m$ et retour à l'étape (b).

4. Calcul de l'indice $\mathbf{k}$ de $\mathbf{y}$ dans Q_n.

La complexité de cet algorithme simplifié n'est pas bornée et dépend de $\mathbf{x}$. Pour vérifier que $\mathbf{y} \in Q_n$, on peut utiliser les algorithmes d'indexation $\mathbf{y} \to \mathbf{k}$, $\mathbf{k} \to \mathbf{y}$, $\mathbf{y} \to (\mathbf{k}, \mathbf{l})$ et $(\mathbf{k}, \mathbf{l}) \to \mathbf{y}$.

5.5.3 Complexité

La complexité de quantification multi-débit est fixée par la partition, $\Lambda/2^r\Lambda$ ou $\Lambda/2^rR\Lambda$, la plus difficile à indexer. En général la partition $\Lambda/2^rR\Lambda$ est plus complexe. En mettant de coté le codage de n, les données à stocker pour réaliser la quantification dans $\Lambda/2^r\Lambda/2^rR\Lambda$ se réduisent

essentiellement à $\mathbf{a}$, $M(\Lambda)$, $M(\Lambda)^{-1}$, $M(R\Lambda)$ et $M(R\Lambda)^{-1}$ ainsi qu'à une matrice décrivant $\Lambda/R\Lambda$.

5.6 Application : quantification vectorielle algébrique multi-débit basée sur RE_8 pour le codage TCX audio

Dans le contexte de l'AMR-WB+, on détaille ici un système de quantification multi-débit alternatif en dimension 8. La quantification multi-débit repose, comme à la section 4.5, sur des dictionnaires de quantification Q_n où $n \in \{0, 2, 3, 4, \cdots, n_{max}\}$. Ici, $n_{max} = 33$.

Le codage est fondé sur les partitions $RE_8/2^r RE_8/2^{r+1}E_8^\star$. Il est réalisé à partir des algorithmes décrits à la section 5.5, en prenant $\Lambda = RE_8$ et $R\Lambda = 2E_8^\star$. Les réseaux RE_8 et $E_8^\star$ sont définis à l'annexe 5.A ; ils sont reliés par des rotations R et $R^\star$ telles que $R \times E_8 = RE_8$ et $R \times RE_8 = 2E_8$, $R^\star \times RE_8 = 2E_8^\star$ et $R^{\star\star} \times E_8^\star = RE_8$.

Le système comprend un codeur et un décodeur, tels qu'illustrés à la figure 5.14. En supposant que l'allocation des bits est suffisante, le vecteur $\mathbf{x}$ est représenté par son plus proche voisin $\mathbf{y}$ dans RE_8. L'indexation multi-débit fait correspondre à $\mathbf{y}$ un numéro de dictionnaire n – identifiant un dictionnaire Q_n – et un indice i représenté sur $4n$ bits. Ce système est structurellement identique à celui présenté à la section 4.5. Les dictionnaires Q_n sont cependant différents.

Le numéro de dictionnaire n est représenté sous forme binaire par codage unaire avant multiplexage.

5.6.1 Décodage multi-débit

Le numéro de dictionnaire n est décodé à partir de sa représentation binaire (un code unaire ici). L'indice i est interprété différemment suivant la valeur de n :

1) Si $n = 0$, le décodeur reconstruit $\mathbf{y} = (0, \cdots, 0)$, le seul élément du dictionnaire de base Q_0.

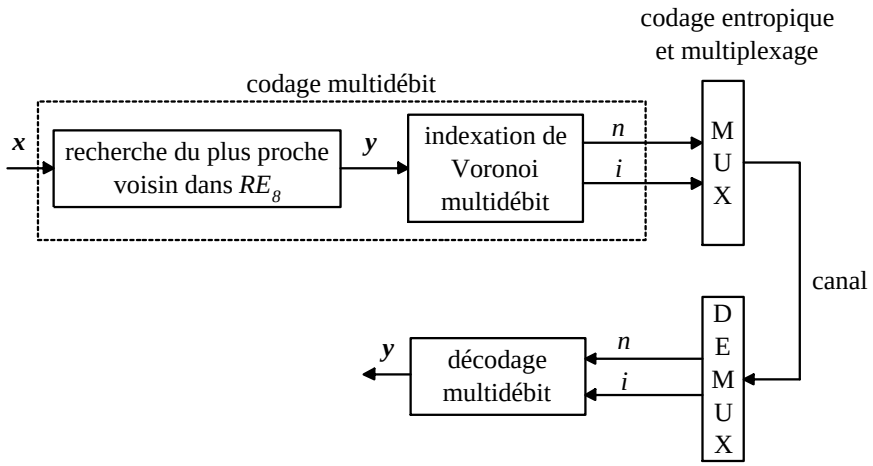


Figure 5.14: Système de quantification multi-débit alternatif basé sur RE_8.

2) Si $n = 2r$ avec r entier ≥ 1, i correspond à un indice $\mathbf{k}$ de Voronoï pour RE_8 sur $8r$ bits, dont les 8 composantes sont comprises entre 0 et $m-1$ avec $m = 2^r$. L'indice $\mathbf{k}$ est décodé en un mot $\mathbf{y}$ de $RE_8/2^r RE_8$ à l'aide de l'algorithme de [4]. Ce décodage est décrit dans les chapitres 3 et 4.

3) Si $n = 2r+1$ avec r entier ≥ 1, l'indice i correspond à deux sous-indices :

   - un indice de Voronoï $\mathbf{k}$ pour $2E_8^\star$ sur $8r$ bits ;
   - un indice $\mathbf{l} = (l_1, l_2, l_3, l_4)$ dont les 4 composantes sont binaires, prenant 4 bits.

   Le décodage de $(\mathbf{k}, \mathbf{l})$ est effectué tel que présenté à la figure 5.15. Le décodage de $RE_8/2^{r+1}E_8^\star$ généralise le décodage de RE_8/mRE_8 de [4]. Il génère toujours un point dans RE_8 car $2mE_8^\star$ est un sous-réseau de RE_8. De plus les points générés sont bien à l'intérieur de la région $2mV(E_8^\star) + \mathbf{a}$.

Le décalage de Voronoï $\mathbf{a}$ est fixé à zéro. Il n'est pas admissible ; cette propriété est prise en compte pendant le codage multi-débit.

Décodage: indice $(\mathbf{k}, \mathbf{l}) \rightarrow$ mot de code $\lambda \in RE_8$

1. Calculer $\mathbf{x} = \mathbf{k}M(2E_8^\star) + \mathbf{l}M(RE_8/2E_8^\star)$
2. Calculer $\mathbf{y} = (\mathbf{x} - \mathbf{a})/(2m)$
3. Trouver le plus proche voisin $\mathbf{z}$ de $\mathbf{y}$ dans $E_8^\star$
4. Calculer $\lambda = \mathbf{x} - 2m\mathbf{z}$

Figure 5.15: Algorithme de décodage de $RE_8/2^{n+1}E_8^\star$ $(m = 2^n)$.

Ainsi, pour $n > 2$, la troncature de RE_8 est $2^r V(RE_8) + \mathbf{a}$ si $n = 2r$, et $2^{r+1}V(E_8^\star) + \mathbf{a}$ si $n = 2r + 1$.

5.6.2 Codage multi-débit

Le codage multi-débit repose essentiellement sur l'algorithme de codage dans $\Lambda/2^r\Lambda/2^r R\Lambda$ détaillé précédemment, avec $\Lambda = RE_8$, $R^\star\Lambda = 2E_8^\star$. Pour r entier ≥ 1,

$$Q_{2r} = RE_8 \cap \{2^r V(RE_8) + \mathbf{a}\} \tag{5.16}$$

et

$$Q_{2r+1} = RE_8 \cap \left\{2^{r+1}V(E_8^\star) + \mathbf{a}\right\}. \tag{5.17}$$

Les étapes de codage sont :

1. Trouver le proche voisin $\mathbf{y}$ de $\mathbf{x}$ dans le réseau infini RE_8.
2. Si $\mathbf{y} = (0, \cdots, 0)$, $n = 0$. Fin.
3. Trouver la troncature minimale de RE_8 pour indexer $\mathbf{y}$ dans Q_n à l'aide des partitions $RE_8/2^r RE_8/2^{r+1}E_8^\star$.
4. Indexer $\mathbf{y}$ dans Q_n :

(a) Si $n = 2r$:

Calculer l'indice $\mathbf{k}$ de Voronoï de $\mathbf{y}$ dans $RE_8/2^r RE_8$ à l'aide de l'algorithme de [4] (voir aussi les chapitres 3 et 4). Si l'opération $\mathbf{k} \rightarrow \mathbf{z}$ donne $\mathbf{z} = \mathbf{y}$, Fin. Sinon incrémenter n de 1 et aller à l'étape (b).

(b) Si $n = 2r + 1$:

L'indexation est effectuée séquentiellement en identifiant d'abord le sous-indice $\mathbf{l}$ puis le sous-indice $\mathbf{k}$. Cette procédure, similaire à la modulation codée de [5], est présentée à la figure 5.16. Si l'opération $(\mathbf{k}, \mathbf{l}) \rightarrow \mathbf{z}$ donne $\mathbf{z} = \mathbf{y}$, Fin. Sinon incrémenter n de 1 à l'étape (a)

Codage: mot de code $\mathbf{y} \in RE_8 \rightarrow$ indice $(\mathbf{k}, \mathbf{l})$

1. $\mathbf{t} = \mathbf{y} M(RE_8)^{-1}$
2. $\mathbf{u} = (t_2 - t_5 + t_7, t_3 - t_5 + t_6, t_4, t_8)$ et $\mathbf{l} = (u_1 \ mod\ 2, \cdots, u_4 \ mod\ 2)$
3. $\mathbf{z} = \mathbf{y} - \mathbf{l} M(RE8/2E_8^\star)$
4. $\mathbf{j} = \mathbf{z} M(2E_8^\star)^{-1}$
5. $\mathbf{k} = (j_1 \ mod\ m, \cdots, j_N \ mod\ m)$

Figure 5.16: Algorithme de codage de $RE_8/2mE_8^\star$ $(m = 2^r)$.

Pour dériver le décodage de $RE_8/2mE_8^\star$, on suppose que $\mathbf{y}$ est généré par décodage de $(\mathbf{k}, \mathbf{l})$. Alors on peut écrire : $\mathbf{y} = \mathbf{k} M(2E_8^\star) + \mathbf{l} M(RE_8/2E_8^\star) - 2m\mathbf{z}$ avec $\mathbf{z} \in E_8^\star$ et $m = 2^r$ (r entier ≥ 1).

L'objectif est alors de retrouver $(\mathbf{k}, \mathbf{l})$ à partir de $\mathbf{y}$. Le résultat de l'étape 1 du décodage est

$$\mathbf{t} = \begin{bmatrix} k_1 - k_2 - k_3 - k_5 - k_4 - k_6 - k_7 \\ 2k_2 + k_5 + k_6 + l_1 \\ 2k_3 + k_5 + k_7 + l_2 \\ 2k_4 + l_3 \\ k_5 + k_6 + k_7 \\ k_6 \\ k_7 \\ 2k_8 + l_4 \end{bmatrix}^t - m(2\mathbf{z})M(RE_8)^{-1} \tag{5.18}$$

Puisque $2E_8^\star$ est un sous-réseau de RE_8 et $m = 2^r$, $2zM(RE_8)^{-1}$ est entier et $m(2\mathbf{z})M(RE_8)^{-1}$ est un multiple de 2. En calculant $\mathbf{u} = (t_2 - t_5 + t_7, t_3 - t_5 + t_6, t_4, t_8)$ et en appliquant un modulo 2 sur les composantes de $\mathbf{u}$, on trouve : $l_i = u_i \; mod\, 2$ pour $1 \leq i \leq 4$. Le vecteur $\mathbf{l}$ étant identifié, la quantité $\mathbf{l}M(RE_8/2E_8^\star)$ peut être retirée de $\mathbf{y}$ afin d'identifier $\mathbf{k}$.

5.6.3 Comparaison avec le système de quantification multi-débit dans RE_8 basé sur l'extension de Voronoï

Performances : cas d'une source gaussienne

On prend ici une source gaussienne i.i.d de moyenne nulle et de variance unité. Celle-ci est codée par blocs de 8 échantillons. Plus précisément, cette source est quantifiée dans gRE_8 où g est un facteur d'échelle servant à contrôler le débit moyen R du codage multi-débit et la distorsion moyenne normalisée D.

L'indexation multi-débit repose sur des dictionnaires Q_n de RE_8 de débit $4n$ bits par vecteur, avec $n \in \{0, 2, 3, 4, 5, \cdots, n_{max}\}$. Les numéros de dictionnaires n sont représentés par codage unaire. On compare les performances des systèmes décrits aux sections 4.5 et 5.6. La différence entre ces systèmes tient à l'indexation multi-débit (ainsi qu'au multiplexage et démultiplexage).

Les résultats de simulation sont présentés à la figure 5.17. À un débit inférieur à 3 bit par dimension, le système développé ici, basé sur le codage de Voronoï à troncature adaptative, a des performances moindres que le système basé sur l'extension de Voronoï. Cet écart s'explique facilement par l'asymétrie et le décentrage des codes de Voronoï. À un débit supérieur à 3 bits par dimension, les systèmes ont des performances équivalentes, avec un avantage négligable pour le système basé sur la troncature de Voronoï adaptative.

Pour améliorer les performances à bas débit, on définit un système hybride. Celui-ci est spécifié par les dictionnaires :

- Q_0 réduit à $(0, \cdots, 0)$;
- Q_2 construit à partir de 3 leaders absolus de RE_8 – le même Q_2 que le système basé sur l'extension de Voronoï décrit à la section 4.5 ;
- pour $n > 2$, Q_n du système décrit à la section 5.6.

Ce système hybride corrige les défauts du codage de Voronoï (asymétrie et décentrage), qui s'avèrent pénalisants à faible débit.

Complexité

Le codage multi-débit comporte 3 phases : la recherche du plus proche voisin $\mathbf{y}$ dans RE_8, puis l'indexation multi-débit revenant à calculer le numéro de dictionnaire n et à indexer $\mathbf{y}$ dans Q_n, et enfin le codage de n. Le décodage multi-débit se réduit au décodage du numéro de dictionnaire n et au décodage de l'indice i de $\mathbf{y}$.

Dans le système décrit à la section 5.6, le décodage dans Q_n ($\mathbf{k} \to \mathbf{y}$ ou $(\mathbf{k}, \mathbf{l}) \to \mathbf{y}$) est réalisé sans stockage par recherche du plus proche voisin dans RE_8 ou $2E_8^\star$. D'après [8], l'opération $\mathbf{k} \to \mathbf{y}$ dans RE_8 est approximativement aussi complexe que le décodage du rang. Par ailleurs, l'opération $(\mathbf{k}, \mathbf{l}) \to \mathbf{y}$ est légèrement plus complexe que $\mathbf{k} \to \mathbf{y}$, car elle repose sur le décodage de $2E_8^\star$, étudié à

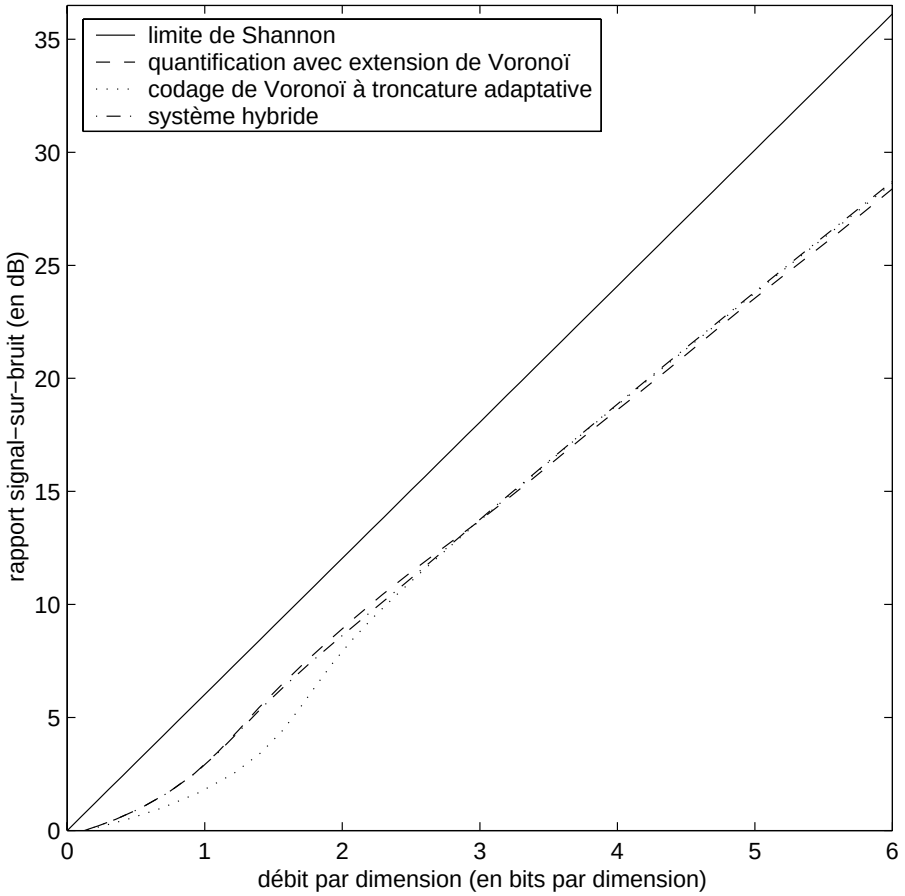


Figure 5.17: Comparaison des systèmes de quantification multi-débit dans RE_8 avec codage unaire de l'information supplémentaire.

l'annexe 5.B. Néanmoins, pour ce qui concerne le décodage, le système basé sur le codage de Voronoï à troncature adaptative est moins complexe que son équivalent basé sur l'extension de Voronoï.

La complexité du décodage est en fait critique dans des applications de type diffusion sur terminaux mobiles. Dans ce contexte, le système décrit à la section 5.6 semble plus intéressant que celui de section 4.5.

5.7 Conclusions

Ce chapitre a été consacré à une technique efficace de quantification vectorielle algébrique multi-débit. La source est représentée dans un réseau régulier Λ infini et le point de Λ le plus proche de la source est indexé de façon adaptative à l'aide du codage de Voronoï. Ce type de codage nécessite de coder une information supplémentaire indiquant la troncature de Voronoï de Λ. Tel qu'évoqué dans [5], cette idée de troncature adaptative vient naturellement en étudiant le codage de Voronoï :

> *Voronoi constellations are also naturally scalable. Since* $|\Lambda/2\Lambda_S| = 2^N|\Lambda/\Lambda_S|$*, a Voronoi constellation based on the partition* $\Lambda/2\Lambda_S$ *can represent* N *more bits per dimensions than one based on* Λ/Λ_S*. When* Λ *and* $2\Lambda_S$ *are "binary lattices," we may often obtain a finer-grained scaling from the partition* $\Lambda/R\Lambda_S$*; since* $\Lambda/R\Lambda_S = 2^{N/2}|\Lambda/\Lambda_S|$*, such partitions allow increments in normalized bit rate of one bit per two dimensions.*

Le codage multi-débit a été développé ici à partir d'une partition $\mathbb{Z}/m\mathbb{Z}$ et plus généralement $\Lambda/m\Lambda$, où Λ est un réseau régulier quelconque en dimension N. La complexité de codage et décodage est indépendante du débit. Pour que les dictionnaires de quantification soient de débit entier par dimension, m doit être une puissance de 2. Pour obtenir des incréments de débit de $\frac{1}{2}$ bit par dimension, le codage doit être effectué dans la partition $\Lambda/2^r\Lambda/2^rR\Lambda$, où Λ est un réseau binaire en dimension paire $N \geq 2$.

Comme la quantification multi-débit par extension de Voronoï étudiée au chapitre 4, le codage par troncature de Voronoï adaptative possède les avantages et inconvénients propres à la quantification multi-débit sans saturation. Il est néanmoins de plus faible complexité que son équivalent basé sur l'extension de Voronoï, en particulier sa complexité est indépendante du débit des dictionnaires. Par contre, il s'appuit sur une contrainte structurelle forte : la forme des dictionnaires est imposée (codes de Voronoï) ; de plus, cette forme n'est efficace que dans le cas où la source vectorielle est gaussienne avec des composantes indépendantes et identiquement distribuées.

Annexe 5.A : Réseaux RE_8, E_8 et $E_8^\star$

Le réseau de Gosset en dimension 8 est habituellement noté E_8 (même pour désigner différentes versions de ce réseau). Pour éviter toute ambiguïté, on définit ici clairement les différentes versions de E_8 utilisées dans ce chapitre.

Les vecteurs sont ici par convention des vecteurs lignes.

Réseaux E_8 et $E_8^\star$

Le réseau E_8 est défini comme le réseau régulier obtenu par construction A [9] à l'aide du code de Hamming étendu $[8, 4, 4]$ binaire spécifié par la matrice génératrice

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \tag{5.19}$$

On a ainsi [3]

$$E_8 = 2\mathbb{Z}^8 + [8, 4, 4]. \tag{5.20}$$

Le réseau E_8 peut être spécifié par la matrice génératrice [9, p. 233] :

$$M(E_8) = \left[\begin{array}{cccccccc} 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}\right] \begin{array}{l} \\ \\ \\ \leftarrow \\ \\ \leftarrow \\ \leftarrow \\ \leftarrow \end{array} \left.\begin{array}{l} \\ \\ \\ \\ \\ \\ \\ \end{array}\right\} \begin{array}{l} \\ \\ \\ \\ \\ \text{partie} \\ \text{[8,4,4]} \\ \\ \end{array} \tag{5.21}$$

Une construction alternative employant le code $[8,4,4]$ dont les symboles sont transposés en ± 1 est donnée dans [2] ; celle-ci conduit à une structure algébrique différente de E_8.

Au lieu de E_8, on emploit ici le réseau $E_8^\star$ spécifié par

$$M(E_8^\star) = \left[\begin{array}{cccccccc} 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}\right] \begin{array}{l} \left.\begin{array}{l} \\ \\ \\ \\ \end{array}\right\} \text{partie } 2\mathbb{Z}^8 \\ \left.\begin{array}{l} \\ \\ \\ \\ \end{array}\right\} \text{partie } [8,4,4]^\star \end{array} \tag{5.22}$$

Ce réseau est généré par construction A avec le code $[8,4,4]$ dont les bits 4 et 5 sont permutés ; cette manipulation revient à permuter les 4-ième et 5-ième composantes de E_8.

Ce réseau $E_8^\star$ ne doit pas être confondu avec le réseau E_8^* (E_8 non-standard) de [5], défini par la matrice génératrice

$$M(E_8^*) = \left[\begin{array}{cccccccc} 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}\right] \begin{array}{l} \left.\begin{array}{l} \\ \\ \\ \\ \end{array}\right\} \text{partie } 2\mathbb{Z}^8 \\ \left.\begin{array}{l} \\ \\ \\ \\ \end{array}\right\} \text{partie } [8,4,4]^* \end{array} \tag{5.23}$$

Réseau RE_8

En suivant la notation de [3], on définit le réseau RE_8 équivalent à E_8 par la matrice génératrice [9] :

$$M(RE_8) = \begin{bmatrix} 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}. \tag{5.24}$$

Le R de RE_8 indique que RE_8 est une version tournée de E_8. Plus précisément, $RE_8 = R \times E_8$, c'est-à-dire que RE_8 est obtenu en multipliant E_8 par une transformation R, avec

$$R = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix}. \tag{5.25}$$

Puisque les vecteurs sont par convention des vecteurs lignes, la multiplication de E_8 par R doit être bien définie avec précaution :

$$R \times E_8 = \left\{ (R\mathbf{x}^t)^t = \mathbf{x}R^t | \mathbf{x} \in E_8 \right\} \tag{5.26}$$

On peut vérifier que $R^2 E_8 = 2E_8$ et montrer que

$$RE_8 = 4\mathbb{Z}^8 + 2[8,7,2] + [8,1,1]. \tag{5.27}$$

En définissant

$$R^\star = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix} \tag{5.28}$$

on peut vérifier que $R^\star \times E_8^\star = RE_8$. De plus, $|RE_8/2E_8^\star| = 2^4$ et

$$RE_8 = 2E_8^\star + M(RE_8/2E_8^\star), \tag{5.29}$$

avec

$$M(RE_8/2E_8^\star) = \begin{bmatrix} 2 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}. \tag{5.30}$$

Enfin $R^\star RE8 = 2E_8^\star$; avec

$$R^{\star\star} = 2R^{\star\,-1} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix} \tag{5.31}$$

on vérifie que $R^{\star\star} \times E_8^\star = RE_8$.

Justification du choix de $E_8^\star$ au lieu de E_8

Le réseau $E_8^\star$ est employé au lieu de E_8 car le codage de $RE_8/2^{r+1}E_8$ est plus complexe que celui de $RE_8/2^{r+1}E_8^\star$.

L'équation 5.18 devient en effet pour E_8 :

$$\mathbf{t} = \begin{bmatrix} k_1 - k_2 - k_3 - k_4 - k_5 - k_6 - k_7 \\ 2k_2 + k_4 + k_6 + l_1 \\ 2k_3 + k_4 + k_7 + l_2 \\ k_4 + l_3 \\ 2k_5 + k_6 + k_7 \\ k_6 \\ k_7 \\ 2k_8 + l_4 \end{bmatrix}^t - m(2\mathbf{z})G_{RE_8}^{-1} \tag{5.32}$$

Il apparaît que la variable k_4 ne peut pas être éliminée par modulo 2 pour identifier l_1, l_2 et l_3. L'identification de $\mathbf{l} = (l_1, l_2, l_3, l_4)$ est possible en testant les deux hypothèses : k_4 pair ou impair.

Annexe 5.B : Algorithmes de recherche dans RE_8 et E_8^*

Recherche du vecteur pertinent dans RE_8 et $E_8^\star$

Il y a 240 vecteurs pertinents pour RE_8 et $E_8^\star$; ces vecteurs correspondent aux vecteurs de norme minimale, donc à la première sphère (orbite) de ces réseaux.

Pour RE_8, ces vecteurs peuvent être énumérés [10] en 112 permutations signées du leader absolu $(2, 2, 0, \cdots, 0)$ et 128 permutations signées du leader $(1, \cdots, 1)$ avec un nombre pair de signes négatifs. La recherche du vecteur pertinent est équivalente à une quantification sphérique et peut être réalisée efficacement d'après [10].

Pour $E_8^\star$, ce réseau étant obtenu par construction A à partir du code de Hamming étendu, la première sphère de $E_8^\star$ comprend [9, p. 121] : 224 permutations signées de $(1, 1, 1, 1, 0, 0, 0, 0)$ où les positions non-nulles définissent un des 14 mots de $[8, 4, 4]$ de poids 4 et 16 permutations signées de $(2, 0, \cdots, 0)$. La recherche du vecteur pertinent peut être réalisée en cherchant la composante maximale en valeur absolue et par décodage à décision souple de $[8, 4, 4]^\star$.

Décodage de RE_8 et $2E_8^\star$: recherche dans le réseau infini

Le décodage de RE_8 se ramène un décodage par cosets [9] – plus précisément des deux cosets de $2D_8$.

Plusieurs méthodes sont comparées ici pour décoder $2E_8^\star$: une recherche dans E_8 après permutation de composantes, une recherche dans RE_8 après rotation par $R^\star$, un décodage direct.

Recherche dans E_8 après permutation de composantes

Le réseau $E_8^\star$ peut être défini comme :

$$E_8^\star = \{(x_1, x_2, x_3, x_5, x_4, x_6, x_7, x_8) | (x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8) \in E_8\}. \tag{5.33}$$

Le décodage de $E_8^\star$ peut donc être réalisé comme ci-dessous :

1. Permuter les composantes 4 et 5 de $\mathbf{x}$;
2. Appliquer le décodage rapide dans E_8 de [3] pour trouver $\mathbf{y}$;
3. Permuter les composantes 4 et 5 de $\mathbf{y}$.

Recherche dans RE_8 après rotation par $R^\star$

Cette méthode exploite le décodage rapide de RE_8 par cosets $2D_8$. Le décodage consiste alors à :

1. Appliquer la matrice de rotation $R^{\star\star}$ sur $\mathbf{x}$ – la matrice $R^{\star\star}$ étant définie à l'équation 5.31 ;
2. Appliquer le décodage dans RE_8 sur le résultat de la rotation ;
3. Appliquer la transformation $R^\star$ sur le résultat du décodage dans RE_8.

Recherche directe par décodage de cosets

Le réseau $E_8^\star$ est défini par la formule :

$$E_8^\star = 2\mathbb{Z}^8 + [8,4,4]^\star, \tag{5.34}$$

où $[8,4,4]^\star$ est une version du code de Hamming non-standard. Cette définition de E_8^* peut être écrite sous la forme :

$$E_8^\star = \bigcup_{\mathbf{c}\in[8,4,4]^\star} 2\mathbb{Z}^8 + \mathbf{c} \tag{5.35}$$

Ainsi $E_8^\star$ comprend 16 cosets de $2\mathbb{Z}^8$.

Étant donné un point quelconque $\mathbf{x}$ dans $\mathbb{R}^8$, le décodage de E_8^* consiste à trouver le plus voisin $\mathbf{y}$ de $\mathbf{x}$ dans E_8^*, soit :

$$\mathbf{y} = \arg\min_{\mathbf{z}\in E_8^*} ||\mathbf{x}-\mathbf{z}||^2. \tag{5.36}$$

Ce plus proche voisin $\mathbf{y}$ est unique si $\mathbf{x}$ n'est pas sur la frontière d'une région de Voronoï de E_8^*. L'approche directe de décodage E_8^* consiste à appliquer un décodage par cosets [2], c'est-à-dire à trouver les 16 candidats correspondant chacun à un coset de $2\mathbb{Z}^8$ dans E_8^*, puis à sélectionner $\mathbf{y}$ parmi ces 16 candidats. Les calculs redondants peuvent être éliminés en calculant les distances de façon récursive (par blocs de 2, 4 puis 8 composantes) ou en appliquant une transformation de Hadamard modifiée.

BIBLIOGRAPHIE

[1] G.D. Forney, "Coset Codes. I. Introduction and Geometrical Classification," *IEEE Trans. on Inf. Th.*, vol. 34, no. 5, pp. 1123–1151, Sep. 1988.

[2] J.H. Conway and N.J.A. Sloane, "Fast quantizing and decoding algorithms for lattice quantizers and codes," *IEEE Trans. Inf. Th.*, vol. IT-28, no. 2, pp. 227–231, March 1982.

[3] G.D. Forney, "Coset Codes. II. Binary Lattices and related codes," *IEEE Trans. on Inf. Th.*, vol. 34, no. 5, pp. 1152–1187, Sep. 1988.

[4] J.H. Conway and N.J.A. Sloane, "A Fast Encoding Method for Lattice Codes and Quantizers," *IEEE Trans. Inform. Th.*, vol. 29, no. 6, pp. 820–824, Nov. 1983.

[5] G.D. Forney, "Multidimensional constellations. II. Voronoi constellations," *IEEE Trans. on Selected Areas in Communications*, vol. 7, no. 6, pp. 941–958, Aug. 1989.

[6] J.-R. Ohm, "Advanced Packet Video Coding Based on Layered VQ and SBC Techniques," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 3, no. 3, pp. 208–221, June 1993.

[7] M. Xie, *Quantification vectorielle algébrique et codage de parole en bande élargie*, Thèse de doctorat, Université de Sherbrooke, Quebec, Canada, Fév. 1996.

[8] F. Labonté, "Étude, optimisation et implémentation d'un quantificateur vectoriel algébrique encastré dans un codeur audio hybride ACELP/TCX," M.S. thesis, Université de Sherbrooke, Québec, Canada, Avril 2001.

[9] J.H. Conway and N.J.A. Sloane, *Sphere Packings, Lattices and Groups*, Springer-Verlag, 3rd edition, 1999.

[10] C. Lamblin, *Quantification vectorielle algébrique sphérique par le réseau de Barnes-Wall - Application au codage de parole*, Thèse de doctorat, Université de Sherbrooke, Québec, Canada, Mars 1988.

Chapitre 6

CODAGE TCX ALGÉBRIQUE MULTI-DÉBIT

Le codage TCX (*Transform-Coded eXcitation*) est une technique de codage audio par blocs, qui combine le codage prédictif et le codage par transformée. À l'instar de [1], cette technique est issue du codage CELP (*Code-Excited Linear Prediction*) ; elle se résume à remplacer la recherche dans le dictionnaire innovateur du codage CELP par un codage direct par transformée d'un signal intermédiaire, appelé *cible* [2].

Ce chapitre traite plus spécifiquement du problème du codage par transformée de la cible TCX. Ce problème est essentiel en codage TCX, car le codage de la cible utilise la majorité du budget de bits et contrôle en grande partie la qualité de la synthèse. On reprend l'approche de codage algébrique introduite par Xie et Adoul dans [3], parce que sa performance a été démontrée dans le cas du codage TCX de parole en bande élargie à 16 kbit/s et qu'elle possède de nombreux avantages. Cette approche algébrique est étendue ici, au cas où la quantification multi-débit est réalisée sans saturation comme aux chapitres 4 et 5.

Un nouvel algorithme de codage de la cible TCX est présenté. La quantification de la cible est en

fait de type gain-forme. Cette technique ne tombe cependant pas dans le cadre de la théorie de [4]. Un *gain*, unique, est optimisé ici dans le domaine logarithmique, en suivant le principe d'allocation des bits par remplissage inverse des eaux (*reverse waterfilling*) [5]. Ce gain contrôle l'allocation des bits et la distorsion, et permet de minimiser l'erreur quadratique sur la cible TCX. La "*forme*" est codée par quantification multi-débit sans saturation par produit cartésien. Le codage algébrique de la cible TCX est ainsi rendu plus flexible ; en particulier, il peut s'adapter à des longueurs de trames et des débits multiples.

L'organisation du chapitre est la suivante. Le modèle TCX est d'abord revu dans la section 6.1. On y rappelle la définition de la cible TCX en distinguant les versions du modèle TCX avec et sans prédiction de pitch, et on analyse les techniques existantes de codage de la cible. Le principe du codage algébrique de [3] est étendu et optimisé à la section 6.2. On détaille ensuite dans la section 6.3 un système complet de codage qui concrétise ce principe dans le contexte du codage ACELP/TCX multi-mode [6]. La performance du codage TCX dans le contexte du codage AMR-WB+ est finalement analysée dans la section 6.4, avant de conclure à la section 6.5.

6.1 Revue du modèle TCX (avec et sans prédiction de pitch)

6.1.1 Historique et intérêt pratique du modèle TCX

L'originalité du codage TCX par rapport aux techniques de codage fréquentiel de [7, 8] tient au fait qu'il applique un codage fréquentiel sur l'excitation du filtre de synthèse LPC (ou résidu de prédiction linéaire), et non sur le signal.

Le codage TCX a été développé initialement par Sanchez et Adoul sous le nom de codage FELP (*Frequency Excited Linear Prediction*), pour le codage bas délai de parole en bande élargie à 24 et 32 kbit/s par trames de 3 ms [9]. Ce codage utilise une prédiction linéaire court-terme de type *backward*, si bien que la prédiction long-terme (ou prédiction de pitch) n'est pas employée [10]. Le modèle FELP a été motivé par le fait qu'à débit élevé, la complexité de la recherche dans le

dictionnaire innovateur du modèle CELP devient prohibitive (voir annexe 6.A). Cette recherche est remplacée par un codage direct en boucle ouverte de la cible du CELP (dans le domaine du signal pondéré) [2]. Par suite, l'intérêt du codage FELP est de pouvoir coder la parole à débit élevé, à partir d'un modèle dérivé du CELP.

Le principe de codage FELP a été repris et adapté par Lefebvre, Salami, Laflamme et Adoul dans [11, 12] pour développer le modèle TCX (*Transform Coded eXcitation*). Ce nom est justifié dans le cas de [11], car le codage par transformée est appliqué cette fois-ci dans le domaine de l'excitation. Par contre, dans [12], la cible TCX est définie dans le domaine du signal pondéré, et le modèle de codage sous-jacent appelé abusivement TCX (comme dans [11]) par extension.

Le codage TCX est appliqué au codage de parole en bande étroite à 8 kbit/s dans [11], et au codage en bande élargie de parole à 16 kbit/s et de musique à 24 kbit/s dans [12]. Dans [11], le signal est codé par trames de 6 ms avec prédiction linéaire *backward* et une prédiction de pitch par trame à partir de la synthèse. Dans [12], la cible TCX est définie dans le domaine du signal pondéré, mais avec une prédiction linéaire *forward* et une prédiction de pitch (optionnelle) à partir de l'excitation recontruite.

Le modèle TCX est en fait équivalent à des techniques de Moreau et Galand [13, 14, 15]. Il a été repris dans [16, 17, 18, 19, 20, 21, 22].

Par rapport aux techniques de codage audio de l'état de l'art (par exemple, MPEG AAC [23, 24, 25] ou AAC+ [26]), l'intérêt principal du codage TCX (sans prédiction de pitch) réside dans le fait qu'il permet de basculer de façon adaptative entre CELP et TCX – c'est-à-dire entre codage temporel et fréquentiel –, afin de coder à la fois la parole et la musique [6].

6.1.2 Codage TCX avec prédiction de pitch

Le principe du modèle TCX avec prédiction de pitch est rappelé à la figure 6.1, comparativement au modèle CELP. Le signal est traité par trames (ou blocs). Pour appliquer la prédiction de pitch, chaque trame est découpée en sous-trames. Les bases du modèle CELP sont rappelées à l'annexe

6.A.

Le problème esssentiel du codage CELP consiste à coder le signal intermédiaire $x(n)$, appelé cible, défini par

$$x(n) = s_w(n) - s_{w0}(n) - s_p(n), \tag{6.1}$$

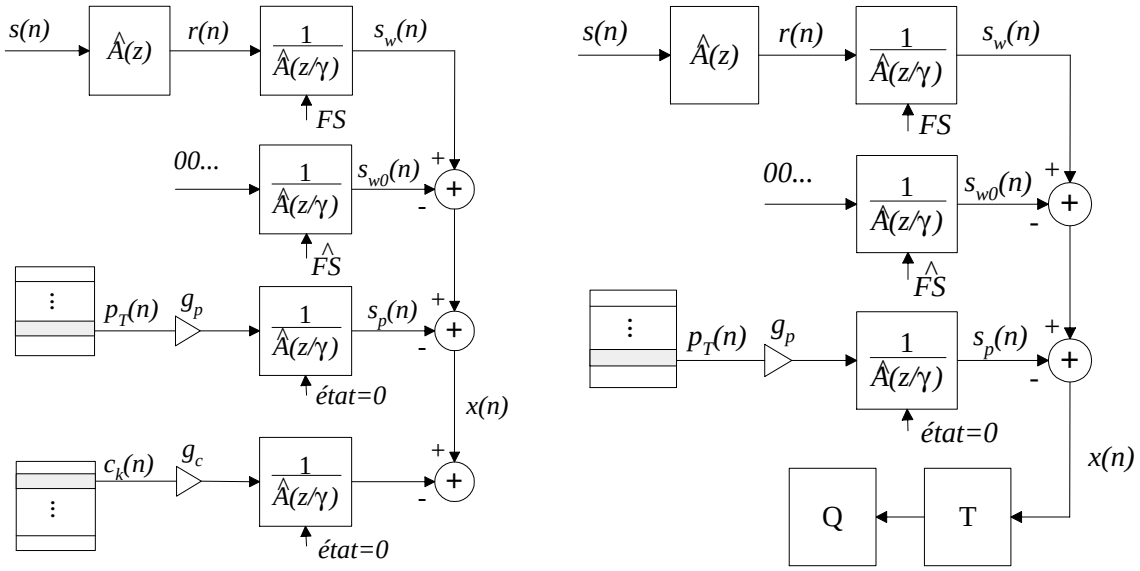
où $s_w(n)$ est le signal original filtré par $W(z)$, $s_{w0}(n)$ est la réponse à une entrée nulle (appelée *ringing* en anglais) du filtre $1/\hat{A}(z/\gamma)$ et $s_p(n)$ est la contribution de pitch (i.e. un mot du dictionnaire adaptatif convolué par la réponse impulsionnelle $h_\gamma(n)$ de $1/\hat{A}(z/\gamma)$ et mis à l'échelle par le gain de pitch g_p). Le filtre de pondération perceptuelle $W(z)$ est fixé à $\hat{A}(z)/\hat{A}(z/\gamma)$. Les mémoires FS et $\widehat{FS}$ représentent l'état du filtre $1/\hat{A}(z/\gamma)$ [27], quand la sortie du filtre est respectivement $s_w(n)$ et $\hat{s}_w(n)$, où $\hat{s}_w(n)$ est la cible reconstruite.

À la différence du CELP qui minimise l'erreur quadratique de modélisation de la cible $x(n)$ au moyen d'un dictionnaire d'excitations filtrées, le modèle TCX représente $x(n)$ par codage direct par transformée, avec un fenêtrage rectangulaire. Alors que le modèle CELP utilise un dictionnaire d'excitations (filtrées), le modèle TCX utilise un dictionnaire de cibles dans le domaine transformé [27]. Le problème de la complexité de la recherche dans le dictionnaire innovateur en boucle fermée du CELP est ainsi contourné par un codage direct de forme d'onde (en boucle ouverte).

Comme le critère de recherche du CELP utilise l'erreur quadratique dans le domaine du signal pondéré (par le filtre $W(z)$), le codage de la cible TCX est aussi effectué suivant l'erreur quadratique dans le domaine du signal pondéré transformé. Le critère de distorsion est donc analytique et bien défini. Les outils de quantification développés en théorie de la distorsion pour l'erreur quadratique [28, 29, 30] s'appliquent idéalement au modèle TCX.

Les paramètres du modèle TCX se réduisent aux coefficients de prédiction linéaire, au délai et au gain du pitch, et aux paramètres de quantification de la cible. La transformation T est généralement orthogonale ; le critère de distorsion étant l'erreur quadratique, l'erreur de quantification $x(n) - \hat{x}(n)$ a un spectre plat en moyenne. Le bruit de codage $s(n) - \hat{s}(n)$ est mis en forme par $W(z)^{-1}$.

A débit élevé, le codage direct dans un domaine transformé (en général fréquentiel) est moins



(a) CELP (b) TCX

Figure 6.1: Comparaison entre CELP et TCX pour un filtre perceptuel $W(z) = \hat{A}(z)/\hat{A}(z/\gamma)$: traitement effectué dans chaque sous-trame.

complexe que l'approche temporelle du CELP. Mais il est également moins efficace à débit comparable que le CELP pour représenter des signaux de parole, car un minimum de bits doivent être alloués aux phases (en supposant que T est une transformée de Fourier discrète) afin de localiser l'énergie dans le temps.

6.1.3 Codage TCX sans prédiction de pitch

Afin de représenter les signaux de musique, la prédiction de pitch a été supprimée du modèle TCX à prédiction linéaire *forward* de [12]. Les paramètres du modèle TCX se réduisent alors aux coefficients de prédiction linéaire et aux paramètres de quantification de la cible. Le codage de la cible TCX par transformée peut être appliqué sur de longues trames (de l'ordre de 20 à 80 ms), afin de maximiser le gain du codage par transformée – ce gain est défini dans [7, 8].

Le codage TCX sans prédiction de pitch, illustré à la figure 6.2, peut être vu comme un codage

par transformée avec fenêtrage rectangulaire dans le domaine pondéré, c'est-à-dire le domaine du signal original filtré par $W(z)$. Il s'agit donc d'une sorte de codage RELP (*Residual-Excited Linear Prediction*) [31] ; cependant, le filtre prédicteur $A(z)$ est remplacé par le filtre perceptuel $W(z)$ et les mémoires de filtre sont prises en compte à travers la mémoire du filtre $W(z)/A(z)$ (ou *ringing*). La figure 6.2 comprend les filtres de pré-accentuation $1 - \mu z^{-1}$ et de désaccentuation $\frac{1}{1-\mu z^{-1}}$.

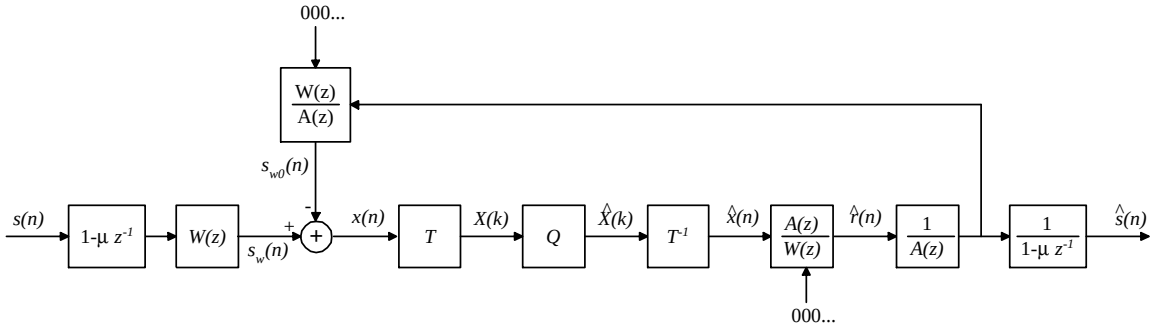


Figure 6.2: Codage TCX sans prédiction de pitch.

6.1.4 Techniques existantes de codage de la cible TCX

On se restreint ici au cas où la transformation orthogonale T est la transformation de Fourier discrète, comme dans [9, 11, 12, 22, 3]. Ce codage fréquentiel permet (en principe) d'exploiter les propriétés fréquentielles de la perception auditive. La théorie du codage de source par transformée de Fourier a été formulée dans [32].

Quantification polaire (amplitude-phase)

Différentes stratégies de codage ont été développées à partir d'une représentation complexe polaire de la cible TCX :

- Dans [9], les amplitudes sont codées par quantification scalaire, alors que les phases sont codées par quantification scalaire ou par quantification par réseaux de points avec troncature cubique

adaptative. La théorie de [32] est appliquée pour répartir le budget de bits entre amplitudes et phases.

- Dans [11], le spectre est d'abord décimé en hautes fréquences (on ne code que 16 raies complexes au lieu de 24) ; le spectre décimé est ensuite normalisé, les amplitudes sont codées par quantification vectorielle non structurée (en un seul bloc) et les phases par quantification dans le réseau Λ_{16} avec troncature cubique adaptative de [9]. La répartition des bits entre amplitudes et phases est optimisée expérimentalement.

- Dans [12, 2], les amplitudes sont codées par sous-vecteurs par quantification vectorielle prédictive de type AR(1). L'allocation des bits aux phases est calculée par algorithme glouton à partir des amplitudes quantifiées (pour chaque bit ajouté, l'amplitude est divisée par 2). Les phases sont codées par quantification scalaire avec décimation [12] ou par quantification vectorielle de la cible complexe normalisée [2]. Typiquement 1/3 des bits sont alloués aux amplitudes et 2/3 aux phases [2].

La stratégie de codage de [12, 2] est illustrée à la figure 6.3. Elle ne minimise pas de façon optimale l'erreur quadratique dans le domaine de la cible transformée $\mathbf{X}$.

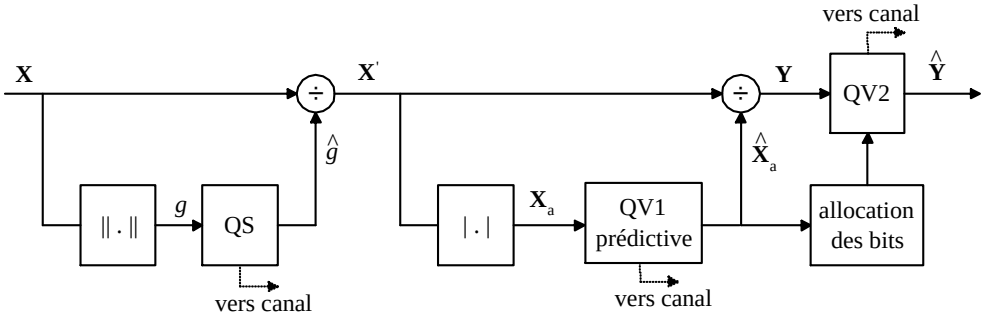


Figure 6.3: Quantification polaire de la cible TCX avec codage prédictif des amplitudes.

Quantification cartésienne (partie réelle-partie imaginaire)

La représentation cartésienne est une représentation alternative. Elle a été adoptée dans [9, 11, 22, 3, 6]. Outre la quantification polaire de [9, 11] :

- Dans [9], les parties réelles et imaginaires du spectre normalisé sont codées par quantification scalaire ou quantification par treillis (TCQ pour *Trellis-Coded Quantization*).
- Dans [11], le spectre est décimé (comme vu précédemment), divisé par sa norme, puis codé par quantification par produit cartésien avec allocation fixe des bits (optimisée expérimentalement).
- Dans [22], le spectre, divisé par un gain choisi parmi plusieurs gains pré-définis, est codé par quantification scalaire avec une allocation des bits (suivant un algorithme de [8]) avec injection de bruit.
- Dans [3], la cible normalisée est codée par quantification vectorielle algébrique. Cette approche a été généralisée dans [6] où plusieurs gains sont testés, avec injection de bruit.

On ne retient ici que le codage algébrique introduit dans [33, chap. 3] et [3] pour le codage de parole en bande élargie à 16 kbit/s. Ce codage est schématisé à la figure 6.4. La cible TCX $x(n)$ est décomposée par transformée de Fourier discrète. Le spectre $X(k)$, comprenant L coefficients transformés, est divisé par sa norme et découpé en $K = L/8$ sous-vecteurs de dimension N =8. Ces sous-vecteurs sont quantifiés individuellement par un même codeur algébrique multi-débit, qui produit un numéro n identifiant un dictionnaire Q_n en dimension 8, ainsi qu'un indice i identifiant un mot de code dans Q_n. Cette quantification est de type gain-forme, et la forme est codée par quantification vectorielle par produit cartésien (en anglais *split vector quantization*). L'allocation des bits est transmise explicitement par l'intermédiaire des numéros de dictionnaire ; celle-ci donne également une représentation approximative de l'énergie dans chaque sous-vecteur.

Dans [3], le gain global du modèle TCX est fixé à la norme du spectre – en suivant le principe de la quantification de type gain-forme. Le nombre de bits à allouer au codeur algébrique multi-débit

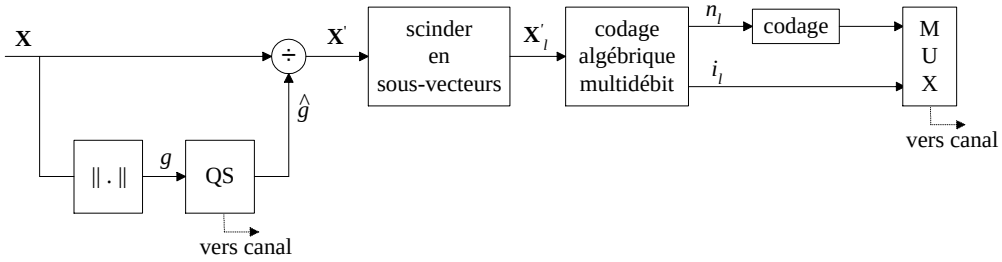


Figure 6.4: Codage algébrique de la cible TCX par quantification vectorielle algébrique imbriquée.

peut donc être majoré en fonction du rapport valeur crête de l'énergie des sous-vecteurs sur énergie totale (*peak-to-mean* en termes de sous-vecteurs). La limitation maximale de l'allocation à 20 bits dans [3] trouve donc une justification, mises à part les considérations de complexité (stockage et saturation). Néanmoins, ce codage rigide n'est pas conçu pour le codage multi-débit (typiquement 16, 24 ou 32 kbit/s en bande élargie) avec différentes longueurs de trames.

6.2 Codage de la cible TCX basé sur la quantification vectorielle algébrique multi-débit sans saturation

On reprend ici l'approche de codage algébrique de [3], suivant laquelle la cible transformée $\mathbf{X}$ est divisée en sous-vecteurs de dimension identique, comme vu au paragraphe précédent. A des fins de généralité, on suppose que $\mathbf{X} = (X_1, \cdots, X_L)$ est de dimension $L = NK$, où K est le nombre de sous-vecteurs de dimension N. On a donc $\mathbf{X} = [\mathbf{X}_1 \cdots \mathbf{X}_K]$, où $\mathbf{X}_k = (X_{Nk-N+1}, \cdots, X_{Nk})$ est le k-ième sous-vecteur. Les composantes scalaires sont ici notées en italiques, tandis que les vecteurs sont notés en gras. Le codage s'appuie sur un réseau de points Λ de dimension N.

A la différence de [3], les sous-vecteurs de $\mathbf{X}$ sont codés ici par quantification vectorielle algébrique multi-débit sans saturation. Celle-ci peut être mise en œuvre au moyen de l'extension de Voronoï ou par codage de Voronoï par troncature adaptative, présentés aux chapitres 4 et 5. La quantification multi-débit sans saturation offre des avantages indéniables pour le codage TCX. Elle est adaptée

au critère de distorsion du codage TCX, car la recherche du plus proche voisin dans un réseau de points est généralement effectuée en minimisant l'erreur quadratique [34]. Elle permet d'allouer virtuellement un nombre arbitraire de bits à chaque sous-vecteur. Cette propriété est essentielle pour coder correctement les sous-vecteurs les plus énergétiques ; autrement, si une erreur est commise sur leur amplitude ou leur phase, on obtient généralement des effet de blocs audibles en bordure de trames [20]. Elle est adaptée au codage par transformée car elle permet de mettre en œuvre des dictionnaires de débits différents et de grande taille. Elle ne requiert virtuellement aucun stockage de dictionnaire et sa complexité algorithmique totale est en général relativement faible par rapport à une quantification vectorielle non-structurée. Elle est enfin plus apte à représenter la variabilité spectrale des signaux audio que des techniques stochastiques (comme celles de [12, 21]) qui sont optimisées pour une source et deviennent défaillantes dès que la source s'écarte du modèle utilisé pendant l'apprentissage (problème d'*outliers*).

6.2.1 Quantification multi-débit sans saturation

Définition générale

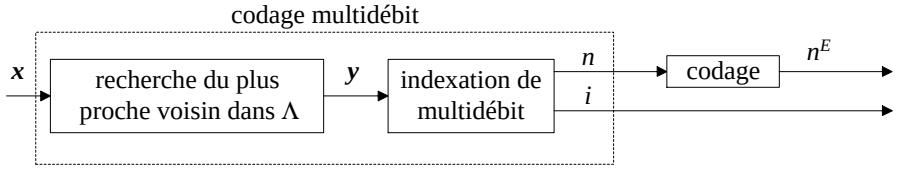
La quantification multi-débit sans saturation est schématisée à la figure 6.5 pour un réseau de points Λ quelconque en dimension N. Cette vue générale englobe à la fois la quantification par extension de Voronoï du chapitre 4 et le codage de Voronoï par troncature adaptative du chapitre 5.

Celle-ci utilise une famille Q_n de dictionnaires de débits distincts, qui sont des sous-ensembles finis du réseau de points Λ (i.e. $Q_n \subset \Lambda$). L'indice n est un numéro de dictionnaire entier ≥ 0 ; en pratique, n est limité par la mise en œuvre à une certaine valeur maximale, qui peut être ajustée de sorte que les dictionnaires Q_n décrivent toute la dynamique de la source.

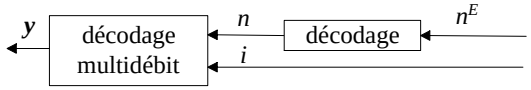
En supposant que le nombre de bits disponibles est suffisant, le codeur représente le vecteur d'entrée $\mathbf{x} \in \mathbb{R}^N$ sans saturation, dans le sens où $\mathbf{x}$ est approximé par son plus proche voisin $\mathbf{y}$ dans le réseau Λ infini, et indexé de façon adéquate comme un mot de code dans un dictionnaire Q_n (de débit minimal). Le numéro de dictionnaire n (entier) est une information supplémentaire qui doit

être codée pour que le décodeur puisse interpréter correctement l'indice i binaire. L'allocation des bits est donnée indirectement par n.

Le numéro de dictionnaire n est entier ; il est représenté sous forme binaire par un code, noté n^E, par un codage adéquat (codage unaire, de Golomb, etc. [35]).



(a) Codeur



(a) Décodeur

Figure 6.5: Quantification vectorielle algébrique multi-débit sans saturation.

On définit Q_0 comme le sous-ensemble de Λ réduit au seul mot nul, i.e. $Q_0 = \{(0, \cdots, 0)\}$. Cette convention permet de gérer facilement les problèmes de dépassement de budget de bits. En effet, dès que le nombre de bits disponibles est insuffisant pour coder $\mathbf{x}$ dans Λ, on peut simplement fixer le numéro de dictionnaire $n = 0$ et ne transmettre aucun indice i. Le codage de $n = 0$ est alors optionnel si par convention le décodeur force $n = 0$ quand le budget de bits devient insuffisant.

Ainsi, la saturation n'a réellement lieu que dans le cas où le nombre de bits disponibles est insuffisant pour coder $\mathbf{x}$, c'est-à-dire pour décrire n^E et i.

Contrôle de l'allocation des bits et de la distorsion

La quantification multi-débit illustrée à la figure 6.5 ne permet pas d'imposer l'allocation des bits ni de contrôler la distorsion de codage. Par définition, la distorsion est donnée par $E[\|\mathbf{x} - \mathbf{y}\|^2]$, où $\mathbf{y}$

est le plus proche voisin de $\mathbf{x}$ dans Λ. De plus, le nombre de bits utilisés pour le codage de $\mathbf{x}$ dépend de $\mathbf{x}$, des dictionnaires Q_n, de la technique de codage utilisée pour représenter n et du budget de bits.

Pour adapter cette quantification au codage par transformée et contrôler le débit et la distorsion, on peut simplement utiliser un facteur d'échelle g au vecteur d'entrée $\mathbf{x}$, comme à la figure 6.6. La quantification multidébit devient de type gain-forme. Le gain g est quantifié à part.

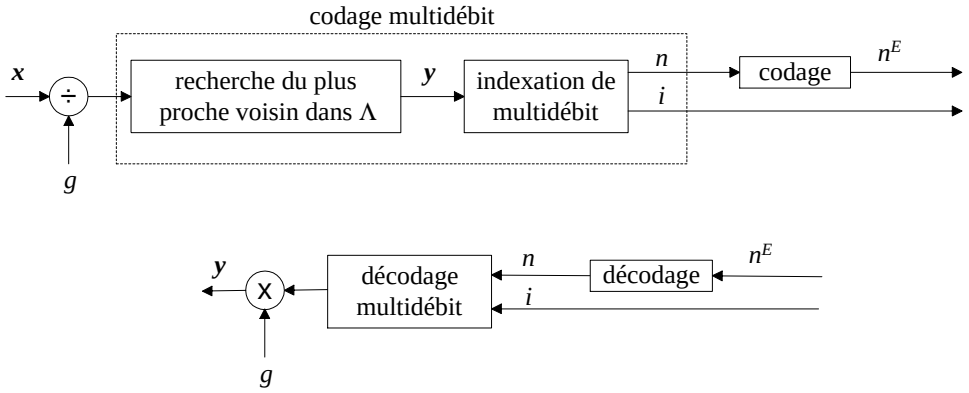


Figure 6.6: Quantification vectorielle algébrique multi-débit sans saturation.

6.2.2 Nouvelle technique de codage de la cible TCX

Principe de codage pour l'erreur quadratique

Le codage de la cible TCX proposé ici est schématisé à la figure 6.7 sous sa forme générale. Un facteur d'échelle est utilisé par sous-vecteur pour contrôler le débit et la distorsion de codage. Dans ces conditions, le problème du codage TCX est essentiellement ramené au contrôle des gains $g_1, \cdots, g_K$. Il ne s'agit pas d'un problème de saturation car celle-ci se réduit ici à un codage dans Q_0.

On rappelle ici que le critère de distorsion du codage de la cible TCX est l'erreur quadratique. Ce critère permet d'appliquer un résultat fondamental de la théorie débit-distorsion : le principe d'allocation des bits par remplissage inverse des eaux (*reverse waterfilling*), applicable et optimal

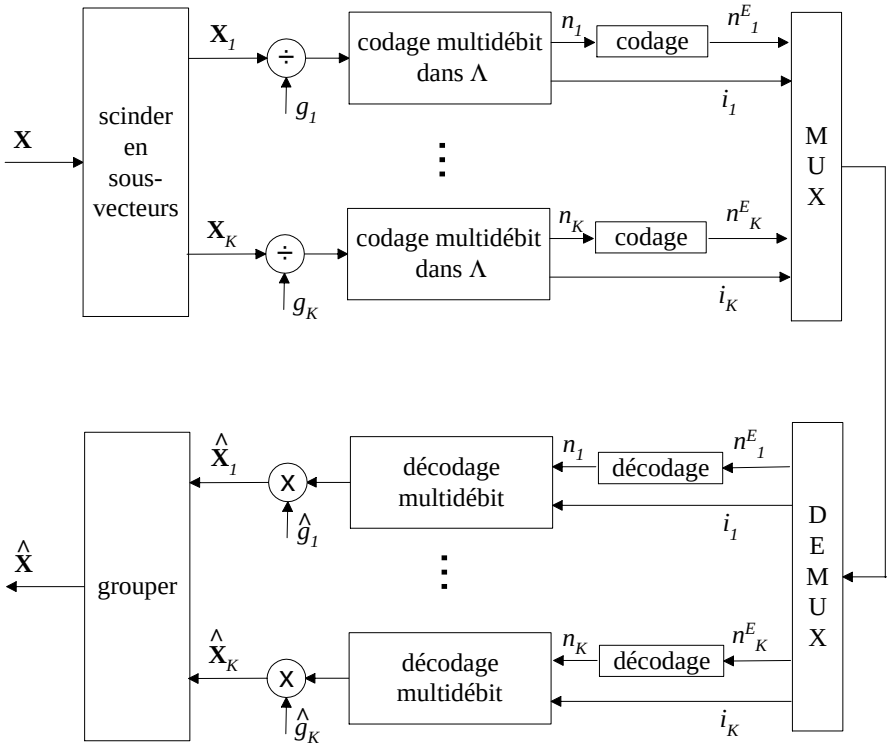


Figure 6.7: Codage de la cible TCX avec quantification multidébit par sous-vecteur.

sous certaines hypothèses – voir annexe 6.B. Ce principe a pour conséquence de simplifier le schéma de la figure 6.7 et l'optimisation des gains $g_1, \cdots, g_K$. En effet, selon ce principe, la meilleure façon d'adapter $g_1, \cdots, g_K$ revient à définir un facteur d'échelle unique g tel que $g_1 = g_2 = \cdots = g_K = g$. On réduit une optimisation multi-variable à une simple optimisation scalaire.

Algorithme général

La cible TCX peut ainsi être codée suivant le schéma présenté à la figure 6.8. On suppose qu'on dispose d'un budget total de B_{tot} bits pour décrire les paramètres de quantification vectorielle algébrique – le codage du gain global n'est pas inclus dans ce budget. Le cible $\mathbf{X}$, de dimension $L = NK$, est codée dans $g\Lambda^K$, où g est un facteur d'échelle (identique pour chaque sous-vecteur)

appelé ici gain TCX. Le problème du codage de la cible TCX est donc ramené essentiellement à l'optimisation du gain global g.

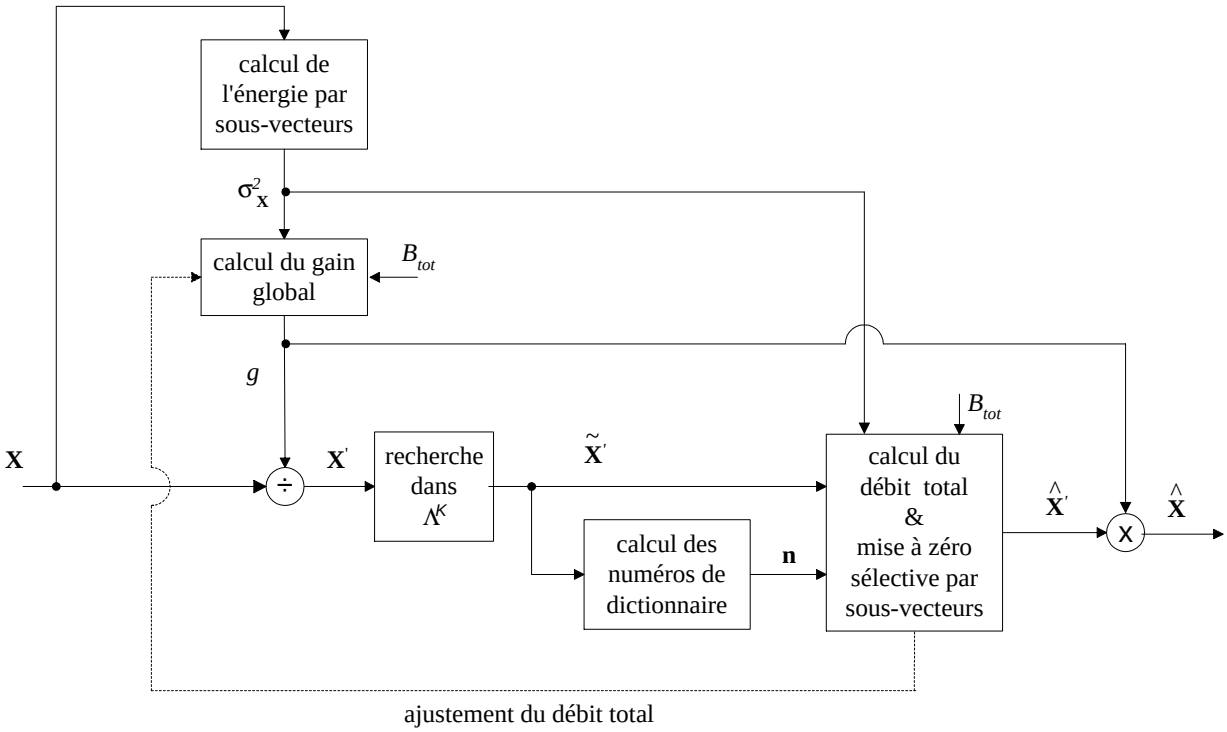


Figure 6.8: Codage optimal de la cible TCX (dans le domaine transformé).

Pour un gain g donné, les étapes de codage permettant de calculer $\hat{\mathbf{X}}$ sont :

1. Normaliser la cible TCX par un gain global g pour obtenir $\mathbf{X}' = \mathbf{X}/g$

2. Trouver le plus proche voisin de la cible normalisée $\mathbf{X}'$ dans Λ^K suivant l'erreur quadratique – le résultat est $\tilde{\mathbf{X}}' = \left[\tilde{\mathbf{X}}'_1 \cdots \tilde{\mathbf{X}}'_K\right]$, où $\tilde{\mathbf{X}}'_k$ est le k-ième sous-vecteur de $\tilde{\mathbf{X}}'$ de dimension N.

3. Pour $k = 1, \cdots, K$, déterminer le numéro de dictionnaire n_k tel que $\tilde{\mathbf{X}}'_k \in Q_{n_k}$ et $\tilde{\mathbf{X}}'_k \notin Q_{n_k-1}$ si $n_k > 0$ – le résultat est $\mathbf{n} = (n_1, \cdots, n_K)$.

4. Calculer la cible normalisée reconstruite $\hat{\mathbf{X}}'$ pour une contrainte du budget de bits B_{tot} par mise

à zéro sélective de sous-vecteurs de $\tilde{\mathbf{X}}'$. Cette mise à zéro est décrite plus loin au paragraphe 6.3.3.

5. Calculer la cible reconstruite $\hat{\mathbf{X}} = g\hat{\mathbf{X}}'$.

Le gain global g contrôle implicitement l'allocation des bits aux sous-vecteurs, ainsi que la distorsion $\|\mathbf{X} - g\hat{\mathbf{X}}'\|^2$; il doit être optimisé pour chaque cible transformée $\mathbf{X}$ et son optimisation est décrite plus loin au paragraphe 6.3.3.

Pour optimiser la performance du codage TCX, le gain, g, n'est pas quantifié directement en boucle ouverte ; il est "re-quantifié". Cette quantification est présentée au paragraphe 6.3.3.

6.3 Quantification de la cible TCX à partir du réseau RE_8 pour le codage AMR-WB+

On décrit ici un système de codage algébrique de la cible TCX basé sur le réseau RE_8 qui étend le système de [3]. On se place ici dans le cas où le réseau est $\Lambda = RE_8$ et la dimension des sous-vecteurs $N = 8$. Cet exemple concret permet de définir plus précisément le principe de codage énoncé précédemment. En particulier on détaille ici un algorithme d'optimisation du gain TCX, qui peut être généralisé à d'autres situations de codage.

Le modèle TCX sous-jacent est présenté succintement à l'annexe 6.C. La longueur des trames peut être fixée à 20, 40 ou 80 ms. Le signal est échantillonné à 16 kHz mais il n'est codé par modèle ACELP/TCX multi-mode que dans une bande couvrant les fréquences 0–6400 Hz – ce même principe de codage de forme d'onde restreint à la bande basse est utilisé dans [36, 26], la limite à 6400 Hz est tirée de [36]. On s'intéresse ici uniquement au bloc de quantification de la cible TCX. L'algorithme de codage correspondant est identique quelque soit le mode TCX choisi. Seuls diffèrent quelques paramètres de l'algorithme : le nombre de sous-vecteurs et le budget de bits.

6.3.1 Système de codage de la cible TCX

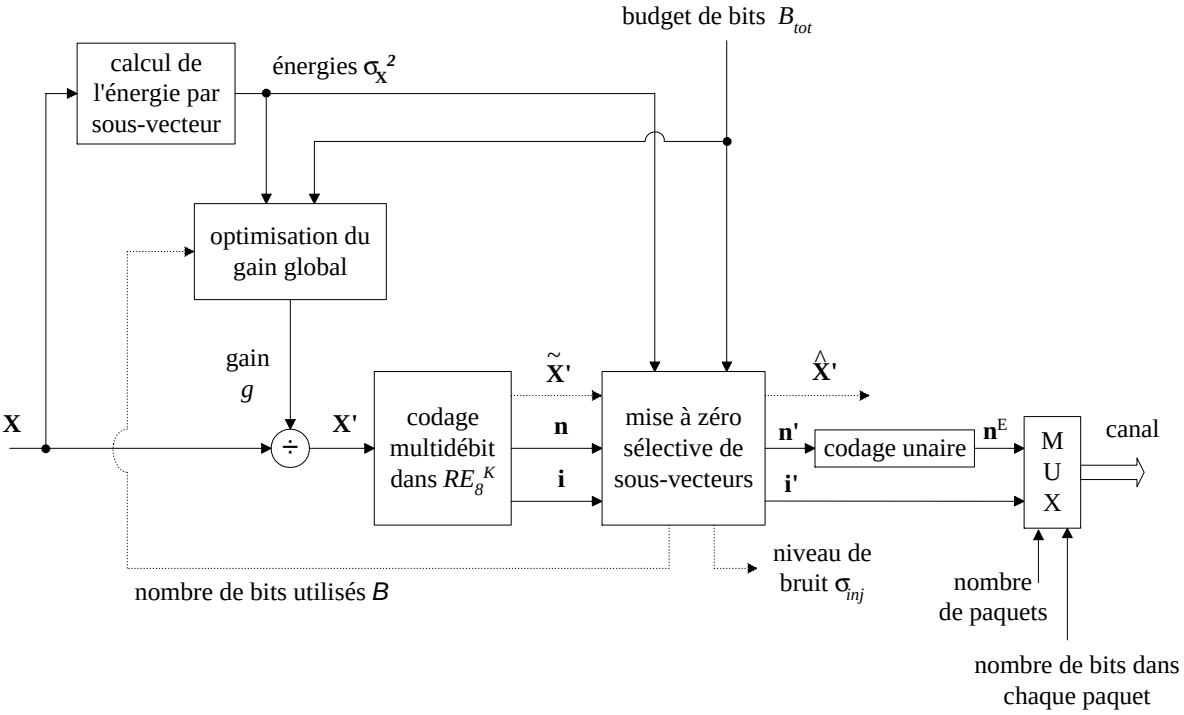
Les codeur et décodeur sont montrés à la figure 6.9. La quantification est de type gain-forme ; le débit de codage et la distorsion sont contrôlés en faisant varier le gain g, appelé gain TCX. Afin de masquer des artefacts de codage, un bruit de confort est injecté au décodeur. Ce bruit n'est introduit qu'à partir d'une certaine fréquence, car l'oreille humaine est très sensible aux défauts en basses fréquences. L'injection de bruit est décrite au paragraphe 6.3.3.

On suppose qu'on dispose d'un budget de B_{tot} bits pour le codage algébrique de la cible TCX. Ce budget n'inclut pas la quantification du gain TCX et du niveau de bruit σ_{inj}. Le gain TCX est codé sur 7 bits avec un pas logarithmique proche de 0.7 dB – on couvre ainsi une dynamique de 90 dB ; celle-ci est suffisante si le format PCM d'entrée a une résolution de 16 bits. Pour économiser des bits, le niveau de bruit est codé relativement au gain TCX sur 3 bits.

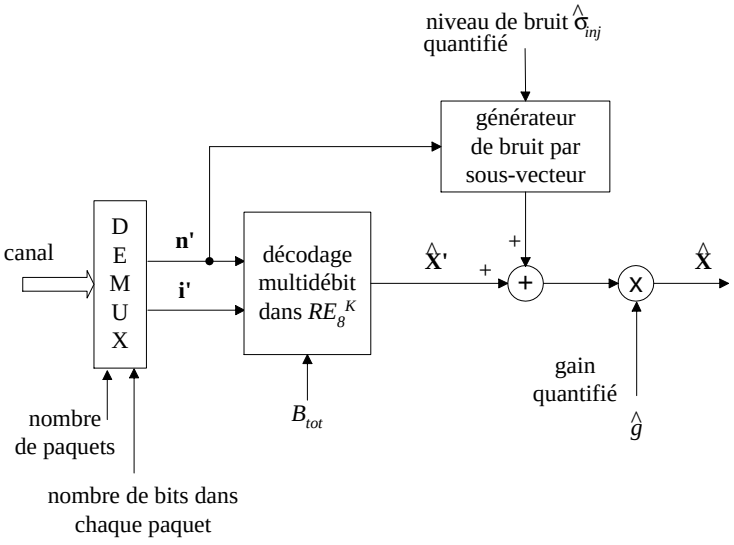
Codeur

Au codeur, le facteur d'échelle $1/g$ est appliqué à la cible transformée $\mathbf{X}$. Le vecteur $\mathbf{X} = (X_1, \cdots, X_L)$ est de dimension $L = 8K$, où K désigne le nombre de sous-vecteurs de dimension 8. Le vecteur $\mathbf{X}' = 1/g\mathbf{X}$ est codé par sous-vecteurs de dimension 8 qui sont quantifiés par RE_8.

On utilise ici une quantification multi-débit sans saturation, qui peut prendre la forme d'un des systèmes de codage dans RE_8 décrits dans les sections 4.5 et 5.6. Le vecteur $\mathbf{X}'$ peut être décomposé sous la forme : $\mathbf{X}' = [\mathbf{X}'_1 \cdots \mathbf{X}'_K]$, où $\mathbf{X}'_k$ est le $k-$ième sous-vecteur défini par $\mathbf{X}'_k = (X'_{8k-7}, \cdots, X'_{8k})$. Chaque sous-vecteur $\mathbf{X}_k$ est codé individuellement. Ce codage produit un numéro de dictionnaire n_k entier ≥ 0 et un indice binaire i_k. Avec les systèmes décrits dans les sections 4.5 et 5.6, i_k est codé sur $4n_k$ bits (par sous-vecteur). Le codage des sous-vecteurs produit donc finalement deux vecteurs : un ensemble de numéros de dictionnaire $\mathbf{n} = (n_1, \cdots, n_K)$ et un ensemble d'indices $\mathbf{i} = (i_1, \cdots, i_K)$. On obtient également une version arrondie de $\tilde{\mathbf{X}}'$ définie comme le plus proche voisin de $\mathbf{X}'$ dans RE_8^K.



(a) codeur



(b) décodeur (en l'absence de pertes de paquets)

Figure 6.9: Codage et décodage de la cible TCX dans l'AMR-WB+.

Puisque la quantification utilisée ne permet par d'imposer une allocation des bits, on est obligé de vérifier *a posteriori* si le nombre de bits utilisés B pour représenter $\mathbf{n}$ et $\mathbf{i}$ respecte le budget de bits B_{tot} alloué. Pour calculer B, les numéros de dictionnaire n_k doivent être mis sous forme binaire. On utilise ici un codage unaire comme spécifié dans les sections 4.5 et 5.6. À l'issue du contrôle du budget de bits, les vecteurs $\tilde{\mathbf{X}}'_k$, $\mathbf{n}$ et $\mathbf{i}$ sont modifiés respectivement en $\hat{\mathbf{X}}'_k$, $\mathbf{n}'$ et $\mathbf{i}$. Le niveau de bruit à injecter est déterminé.

Le gain TCX, g, est optimisé de sorte que le nombre de bits utilisés B corresponde au mieux au budget de bits B_{tot}. L'optimisation obéit au principe du remplissage inverse des eaux et requiert de calculer les sous-vecteurs $\mathbf{X}_k$ par énérgie. La quantification de g est omise à la figure 6.9 ; on suppose que g est re-quantifié.

Décodeur

Au décodeur, en l'absence de pertes de trames et d'erreurs binaires, les numéros de dictionnaire (codés) $\mathbf{n}'$ et les indices $\mathbf{i}'$ sont interprétés pour reproduire le vecteur $\hat{\mathbf{X}}'$. Un bruit de confort complexe (défini dans $\mathbb{C}$) est généré dans chaque sous-vecteur reconstruit dans Q_0, i.e. si $n'_k = 0$.

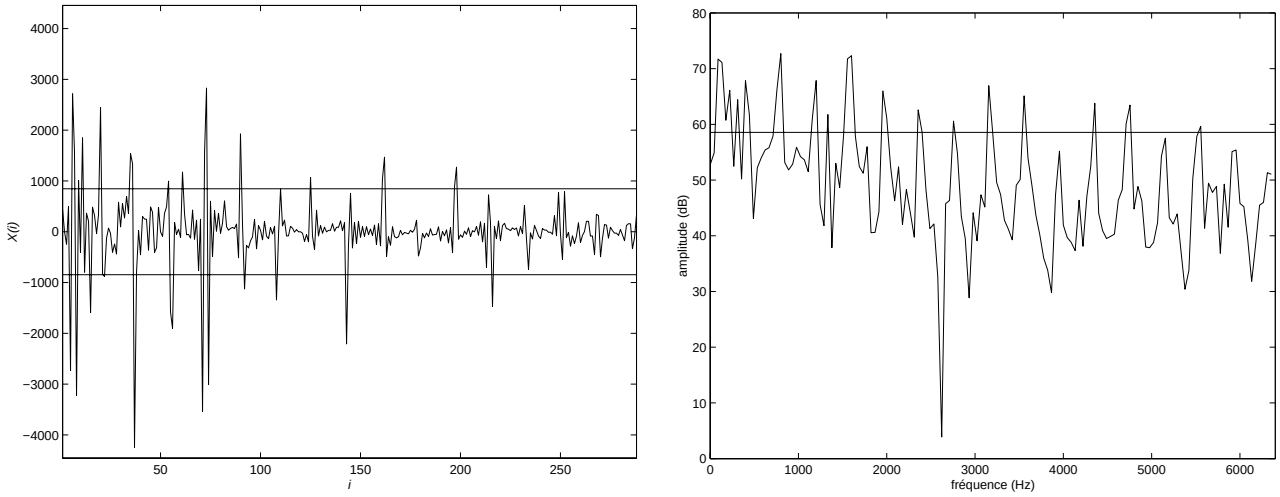
Le vecteur $\hat{\mathbf{X}}'$, auquel du bruit de confort a été ajouté, est finalement multiplié par le gain TCX quantifié $\hat{g}$ pour obtenir la cible reconstruite $\hat{\mathbf{X}}$.

6.3.2 Exemple

Pour rendre les figures plus lisibles, on considère ici un exemple de codage TCX avec une longueur de trame courte, fixée à 20 ms. Tel qu'expliqué à l'annexe 6.C, le vecteur $\mathbf{X}$ contient L=288 coefficients et K=36 sous-vecteurs de dimension 8. Le débit du codage est fixé à 16 kbit/s. Pour ce débit, on fixe le budget de bits B_{tot} à 262 bits. On rappelle qu'en plus des B_{tot} bits, le codage par transformée de la cible TCX consomme 7 et 3 bits pour coder respectivement le gain TCX g et le niveau de bruit σ_{inj}, ainsi que les bits utilisés pour le codage des coefficients LPC. En moyenne, la cible TCX transformée est donc codée à un peu moins de 1 bit par dimension (pour un signal original échantillonné à 16

kHz).

Un exemple de cible TCX transformée est montré à la figure 6.10. Le vecteur $\mathbf{X} = (X_1, \cdots, X_L)$ à la figure 6.10 (a) correspond au spectre complexe de la cible. Comme expliqué à l'annexe 6.C, X_1 est associé à la composante continue (DC), X_2 à la fréquence normalisée $\pi/2$, pour le reste X_{2i}, X_{2i+1} sont respectivement les parties réelles et imaginaires d'une raie de Fourier. Les lignes horizontales représentent le gain global g (et son opposé $-g$) appliqué à $\mathbf{X}$. Le spectre d'amplitude associé se trouve à la figure 6.10 (b). La ligne horizontale représente g en dB. Le calcul de g est expliqué qualitativement ci-après.



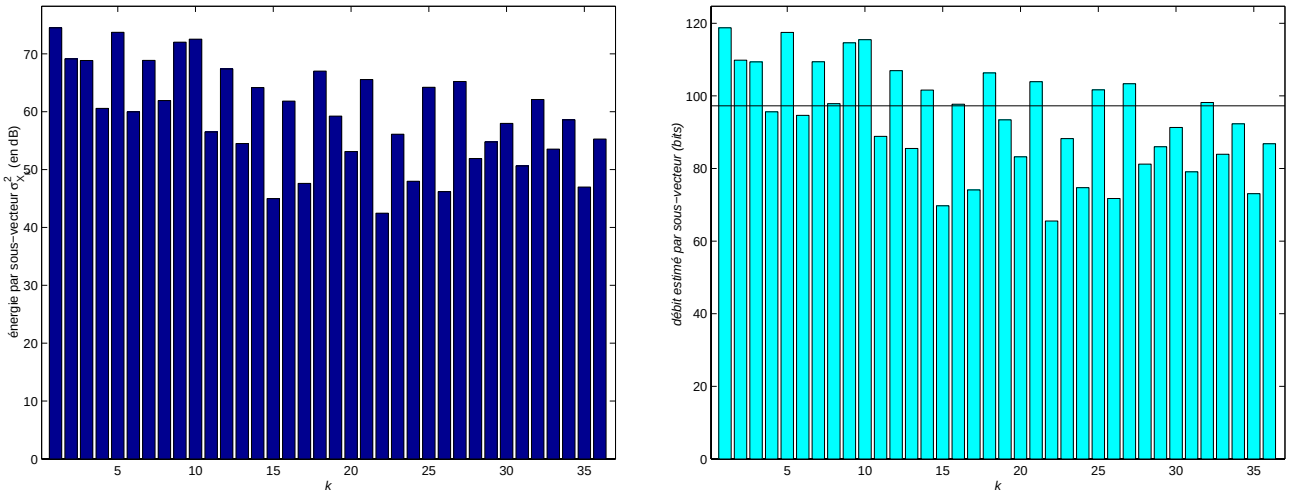
(a) Vecteur $\mathbf{X}$ (b) Spectre d'amplitude (en dB)

Figure 6.10: Cible TCX originale.

L'énergie par sous-vecteur $\sigma_{\mathbf{X}^2} = (\sigma^2_{\mathbf{X}_1}, \cdots, \sigma^2_{\mathbf{X}_K})$, définie par :

$$\sigma^2_{\mathbf{X}_k} = \mathbf{X}_k \mathbf{X}_k^t = X(8k-7)^2 + \cdots + X(8k)^2. \tag{6.2}$$

est représentée à la figure 6.11 (a). À partir de cette énergie, on peut estimer le nombre de bits nécessaires pour coder chacun sous-vecteurs X_k, en supposant que $g = 1$. Cette estimation est montrée à la figure 6.11 (b). La ligne horizontale indique un seuil qui permet d'ajuster g afin de respecter le budget de bits B_{tot}. A priori un sous-vecteur est codé (des bits lui sont alloués), si le nombre de bits estimé correspondant est au dessus du seuil.

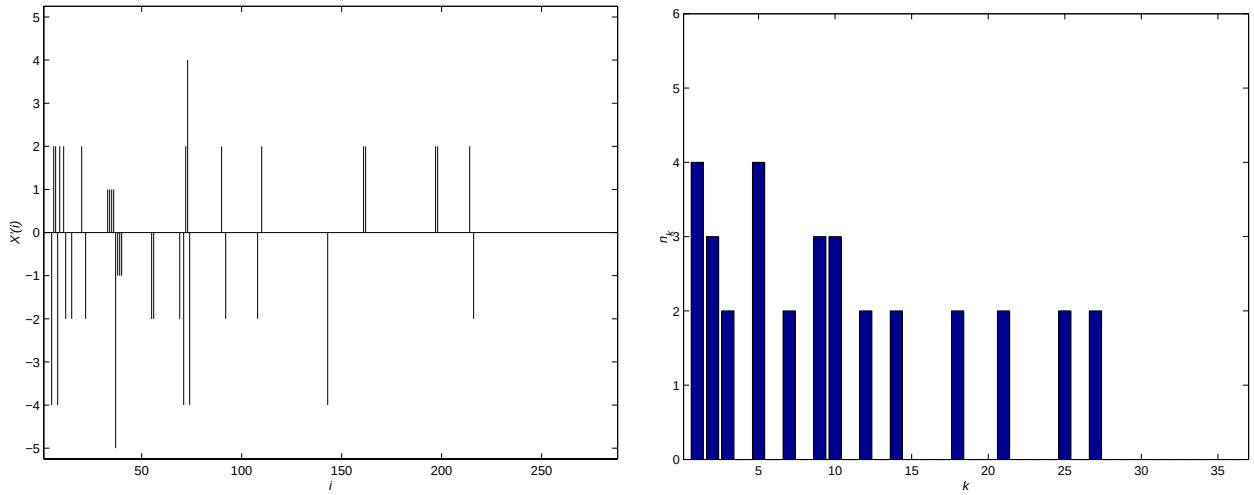


(a) Énergie par sous-vecteur $\sigma^2_{\mathbf{X}_k}$ (b) Allocation estimée des bits dans le cas où $g = 1$

Figure 6.11: Calcul de l'énergie par sous-vecteur et estimation du nombre de bits utilisés.

Le seuil défini à la figure 6.11 peut être associé à une valeur de gain g. Une fois g déterminé, on code le vecteur $\mathbf{X}' = 1/g\,\mathbf{X}$ dans RE_8^K. Le vecteur résultant $\tilde{\mathbf{X}}_k$ est montré à la figure 6.12 (a). L'indexation multi-débit des sous-vecteurs $\tilde{\mathbf{X}}_k$ produit des numéros de dictionnaire n_k qui sont représentés à la figure 6.12 (b). On rappelle qu'avec les systèmes décrits dans les sections 4.5 et 5.6, les dictionnaires Q_n ont un débit de $4n$ bits par dimension ; de plus chaque numéro de dictionnaire n est défini dans $\{0, 2, 3, 4, 5, \cdots, n_{max}\}$. Le dictionnaire Q_1 n'est pas employé pour améliorer l'efficacité de codage : il est en effet peu efficace de coder un sous-vecteurs de dimension 8 avec 4 bits, par rapport à une injection de bruit. Puisque le débit de codage de la cible TCX est inférieur à 1 bit par dimension, une grande partie des sous-vecteurs sont codés dans Q_0, conduisant à $n_k = 0$.

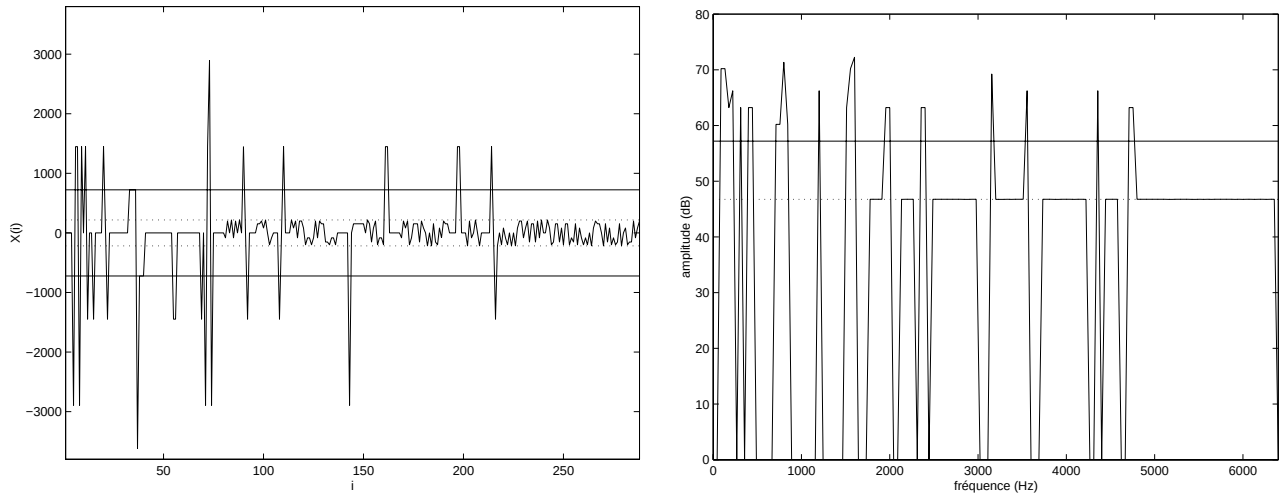
Dans l'exemple choisi, le budget de bits est respecté, si bien que $\hat{\mathbf{X}}' = \tilde{\mathbf{X}}'$, $\mathbf{n}' = \mathbf{n}$, et $\mathbf{i}' = \mathbf{i}$. Au décodeur, du bruit de confort est ajouté au vecteur $\hat{\mathbf{X}}$, puis le gain TCX quantifié $\hat{g}$ appliqué. Le vecteur $\hat{\mathbf{X}}$ – représentant la cible complexe reconstruite – est montré à la figure 6.13 (a). Les lignes horizontales en trait plein représentent $\pm\hat{g}$ (le gain TCX quantifié) ; celles en pointillé correspondent à $\pm\sigma_{inj}$ (le niveau de bruit de confort). Le spectre d'amplitude associé est montré à la figure 6.13 (b).



(a) Vecteur $\tilde{\mathbf{X}}'$ (ici $\hat{\mathbf{X}}' = \tilde{\mathbf{X}}'$) (b) Numéros de quantificateur n_k (ici $n'_k = n_k$)

Figure 6.12: Numéros de quantificateur $\mathbf{n}$ et résultat du codage dans RE_8^K.

Le bruit de confort a un spectre plat (dans le domaine de la cible).



(a) Vecteur $\hat{\mathbf{X}}$ (b) Spectre d'amplitude associé (en dB)

Figure 6.13: Cible TCX reconstruite.

6.3.3 Détails des blocs

On détaille ici chacun des blocs de codage et décodage définis à la figure 6.9.

Codage multi-débit dans RE_8^K

Les sous-vecteurs $\mathbf{X}'_k$ sont codés séparément dans RE_8. La quantification multi-débit dans RE_8 de chaque sous-vecteur $\mathbf{X}'_k$ produit un numéro de dictionnaire n_k et un indice i_k.

Codage unaire de l'information supplémentaire

Pour réduire le débit moyen, n_k et i_k doivent être représentés par codage entropique. Néanmoins, le codage entropique de i_k est difficile à réaliser lorsque i_k a une taille $4n_k \geq 8$ bits. Mis à part Q_0, les dictionnaires Q_n se prêtent donc mal à un tel codage. On ne code donc ici que n_k. Néanmoins, il est en général difficile d'obtenir des statistiques sur n_k car les grandes valeurs de n_k sont peu ou pas du tout utilisées en pratique. Pour simplifier la mise en œuvre, on applique un codage unaire [37, 35]. La fonction de codage associée est donnée au tableau 6.1. Ce code est un code de Huffman optimal pour une distribution de probabilité $p(n) = 2^{-(i+1)}$ avec $i = 0$ si $n = 0$ et $i = n - 1$ si $n > 2$. Il a l'avantage d'être très simple à mettre en œuvre et de faciliter le calcul du nombre de bits utilisés : 1 bit est écrit si $n = 0$, $5n$ bits si $n \geq 2$.

TABLEAU 6.1: Codage unaire et nombre de bits.

Qn	numéro de dictionnaire n	code unaire	nombre de bits pour i	nombre de bits par sous-vecteur
Q0	0	0	0	1
Q2	2	10	8	10
Q3	3	110	12	15
Q4	4	1110	16	20
Q5	5	11110	20	25
⋮	⋮	⋮	⋮	⋮

Calcul de l'énergie par sous-vecteur

L'énergie des sous-vecteurs $\sigma^2_{\mathbf{X}}$ joue un rôle crucial dans le codage algébrique développé ici. Elle contrôle l'allocation estimée des bits (initiale) et sert à initialiser le gain TCX. Son calcul est défini à l'équation 6.2.

Optimisation du gain TCX

Le gain TCX g contrôle implicitement l'allocation des bits aux sous-vecteurs. Il est ici utilisé pour réaliser une optimisation débit-distorsion suivant le principe du remplissage inverse des eaux.

Le gain g peut être initialisé à partir de l'énergie $\sigma^2_{\mathbf{X}_k}$ de chaque sous-vecteur $\mathbf{X}_k$, sans quantification ni calcul du nombre total de bits réellement utilisés pour coder les sous-vecteurs $\mathbf{X}'_k$.

Estimation du nombre de bits par sous-vecteur pour un gain $g = 1$:

Pour un gain $g = 1$ et avec la quantification multi-débit utilisée ici, on peut estimer le nombre de bits utilisés pour coder $\mathbf{X}'_k$ à :

$$b_k = 5 \underbrace{\log_2 \left[(\epsilon + \sigma^2_{\mathbf{X}_k})/2\right]}_{\text{estimation de } n_k}. \tag{6.3}$$

La constante $\epsilon > 0$ sert à éviter le calcul du logarithme de 0 ; elle est en général négligeable par rapport à $\sigma^2_{\mathbf{X}_k}$.

Le calcul de b_k est justifié par les propriétés du réseau RE_8, des dictionnaires Q_n utilisés et du codage unaire :

- Pour un numéro de dictionnaire $n_k \geq 2$, le nombre de bit utilisés est $5n_k$. Cette propriété explique le facteur 5 devant $\log_2 \left[(\epsilon + \sigma^2_{\mathbf{X}_k})/2\right]$.
- La quantité $\log_2 \left[(\epsilon + \sigma^2_{\mathbf{X}_k})/2\right]$ est une estimation du numéro de dictionnaire n_k. Le logarithme $\log_2$ est justifié par le fait que l'énergie moyenne de Q_{n+1} est double de celle de Q_n.

- Le facteur 1/2 appliqué $\epsilon + \sigma^2_{\mathbf{X}_k}$ sert à calibrer l'estimation de n_k pour Q_2. Dans Q_2, l'énergie moyenne est 8, donc $\log_2 \left[(\epsilon + \sigma^2_{\mathbf{X}_k})/2\right] = \log_2 \left[(2+8)/2\right] \approx 2$.

Cette estimation est à la fois valable pour la quantification multi-débit par extension de Voronoï ou par troncature de Voronoï adaptative.

Estimation du nombre de bits par sous-vecteur pour un gain $g > 1$ quelconque :

Quand un gain g quelconque est appliqué au sous-vecteur $\mathbf{X}_k$, l'énergie de $\mathbf{X}'_k = 1/g\mathbf{X}_k$ est obtenue en divisant $\sigma^2_{\mathbf{X}_k}$ par g^2. Le nombre de bits utilisés pour $1/g\mathbf{X}_k$ peut donc être estimé à partir de b_k en soustrayant $5 \log_2 g^2 = 10 \log_2 g$ de b_k :

$$b'_k = b_k - 10 \log_2 g \tag{6.4}$$

Cette estimation est modifiée en pratique pour éliminer les allocations de bits estimées négatives :

$$b'_k = max\left(b_k - 10 \log_2 g, 0\right) \tag{6.5}$$

Estimation du nombre de bits total pour coder $\mathbf{X}'$ et du gain g optimal : L'estimation du nombre total de bits utilisés pour coder $\tilde{\mathbf{X}}'$ vaut donc pour un gain g fixé :

$$B = b'_1 + \cdots + b'_K. \tag{6.6}$$

C'est une fonction non-linéaire de g. Elle est décroissante en fonction de g.

Si un budget de bits B_{tot} est alloué à la quantification multi-débit de $\mathbf{X}$, la valeur optimale de g peut alors être estimée en cherchant g tel que $B = B_{tot}$. Ce problème peut être facilement résolu par un algorithme itératif décrit ci-dessous :

1. Calculer le nombre de bits estimés b_k pour chaque sous-vecteur, suivant l'équation 6.3.

   Ce calcul est nécessaire car, d'après le principe du remplissage inverse des eaux, le gain g doit être calculé à partir de $\log_2 \sigma^2_{\mathbf{X}_k}$ qui est reliée à b_k et les sous-vecteurs doivent être triés suivant $\sigma^2_{\mathbf{X}_k}$ (ou de façon équivalente suivant b_k).

2. Trier les sous-vecteurs par énergie. On trouve ainsi $(l_1, \cdots, l_K)$ tel que $\sigma^2_{\mathbf{X}_{l_1}} \geq \cdots \geq \sigma^2_{\mathbf{X}_{l_K}}$.

3. Estimer le gain g optimal. Cette estimation revient à trouver un λ et un indice de sous-vecteur k tels que

$$\sum_{i=1}^{k} [b_{l_k} - \lambda] \leq B_{tot}. \tag{6.7}$$

L'optimisation de g est donc effectuée dans le domaine de b_k (logarithmique) via le paramètre $\lambda = 10 \log_2 g$.

Cette étape s'explique facilement à partir de l'équation 6.7 :

- initialiser $k = 1$ et $t = b_{l_1}$
- tant que $k < K$ et $(t - kb_{l_{k+1}}) < B_{tot}$
  - $k := k + 1$
  - $t := t + b_k$
- calculer $\lambda = (t - B_{tot})/k$
- trouver g à partir de λ: $g = 2^{\lambda/10}$

Alternativement, une simple recherche par dichotomie peut être développée ; à chaque itération, l'intervalle de recherche est réduit par dichotomie en partant d'un intervalle assez large (en dB).

Optimisation :

L'initialisation de g n'est qu'une estimation de la valeur optimale de g. En effet, elle repose sur une estimation du nombre de bits par sous-vecteur à partir de l'énergie $\sigma^2_{\mathbf{X}_k}$. Le codage peut être amélioré en testant différentes valeurs de g autour de cette valeur initiale, comme dans le codage par transformée de [22].

Contrôle du budget de bits par mise à zéro sélective de sous-vecteurs

Les sous-vecteurs de $\mathbf{X}'$ sont codés individuellement par quantification multi-débit, une fois le gain global g estimé, sous la forme d'un numéro de dictionnaire n_k et d'un indice i_k. Parce que ce codage

multi-débit est effectué en parallèle et indépendamment dans chaque sous-vecteur, le nombre total de bits réellement utilisés peut dépasser ou sous-utiliser le budget de bits B_{tot} disponibles (on parle en anglais respectivement de *budget overflow* ou de *budget underflow*).

La sous-utilisation du budget de bits ne pose pas de problèmes de multiplexage ; dans ce cas, les bits inutilisés sont mis à zéro au moment du multiplexage. La stratégie utilisée ici pour gérer les dépassements de budget de bits consiste à mettre certains sous-vecteurs à zéro de façon sélective (donc à les forcer dans le dictionnaire Q_0). Néanmoins, pour minimiser l'impact sur la distorsion et en accord avec le principe du remplissage inverse des eaux, cette mise à zéro doit être appliquée en priorité aux sous-vecteurs de plus faible énergie.

L'algorithme de contrôle du budget de bits prend ainsi logiquement la forme suivante :

1. Trier les K sous-vecteurs par ordre d'énergie $\sigma^2_{\mathbf{X}_k}$ décroissante – le résultat $(l_1, \cdots, l_K)$ est tel que $\sigma^2_{\mathbf{X}_{l_1}} \geq \cdots \geq \sigma^2_{\mathbf{X}_{l_K}}$. Fixer $l_{max} = 0$.
2. Initialiser le nombre de bits inutilisés B' : $B' = B_{tot}$
3. Pour $k = 1$ à K
   - Calculer le nombre *minimal* de bits b pour décrire le l_k-ième sous-vecteur – pour un codage unaire et la quantification multi-débit dans RE_8 utilisée ici :

     si $n_{l_k} = 0$, $b = 0$

     si $n_{l_k} \geq 2$, $b = 5n_k - 1$
   - Calculer le nombre de bits stop du codage unaire nécessaire : si $l_k > l_{max}$, $l_{max} = l_k$
   - Si $(b + l_{max}) > B'$,

     forcer $n'_{l_k} = 0$ – l'indice i_{l_k} devient inutile

     sinon

     $n'_{l_k} = n_{l_k}$ et $i'_{l_k} = i_{l_k}$

     $B' := B' - b$

La valeur de l_{max} donne le nombre de bits stop qui sont nécessaires pour un décodage correct de la séquence $[n_1^E\ n_2^E\ \cdots\ n_K^E]$.

Multiplexage des paramètres de quantification multi-débit

Le multiplexage consiste à écrire dans un tableau binaire *tab*, de taille B_{tot} égale à l'allocation des bits, les paramètres de la quantification multi-débit, c'est-à-dire :

- Les numéros de dictionnaire encodés $\mathbf{n}^E = (n_1^E, \cdots, n_K^E)$ obtenus à partir de $\mathbf{n}' = (n'_1, \cdots, n'_K)$;
- Les indices des sous-vecteurs $\mathbf{i}' = (i'_1, \cdots, i'_K)$.

On rappelle qu'ici la taille de chaque indice i_k est $4n'_k$ bits. De plus, les numéros de dictionnaire sont définis dans l'ensemble $\{0, 2, 3, 4, \cdots, n_{max}\}$.

Dans le tableau *tab*, le k-ième bit est $tab(k)$; *tab* est indexé des positions 1 à B_{tot}. Les bits des paramètres peuvent être écrits séquentiellement sous la forme :

$$\left[n_1^E\ \ i_1\ \ n_2^E\ \ i_2\ \ n_3^E\ \ i_3 \cdots\right]$$

On adopte ici un autre format :

$$\left[i_1\ \ i_2\ \ i_3 \cdots n_3^E\ \ n_2^E\ \ n_1^E\right]$$

De cette façon, l'écriture est toujours séquentielle (sous-vecteur par sous-vecteur) mais les numéros de dictionnaire et les indices ne sont pas mélangés : les uns sont écrits à partir de la position 1 par position croissante, les autres à partir de la position B_{tot} par position décroissante. Ce format permet de tenir compte de la sensibilité plus importante des numéros de dictionnaire aux erreurs binaires, par rapport aux indices. Avec ce format et pour la quantification multi-débit utilisée, environ 20% des bits sont affectés aux numéros de dictionnaire.

Le multiplexage est donc mis en œuvre ici avec deux pointeurs (ou indices dans *tab*) : un pointeur pos_n indiquant la position actuelle à partir de laquelle un numéro de dictionnaire peut être écrit et

un pointeur pos_i pour les indices. Le pointeur pos_n est initialisé à B_{tot} et pos_i à 1. Les incréments sont négatifs sur pos_n et positifs sur pos_i. A un instant donné, le nombre de bits inutilisés est $pos_n - pos_i + 1$.

Les valeurs du tableau *tab* sont initialisées à zéro. Cette initialisation est liée à la définition du codage unaire. Elle garantit que les bits inutilisés, mis à zéro, correspondent à des numéros de dictionnaire nuls. Elle fait office de bourrage à zéro (*frame fill*). Les sous-vecteurs sont écrits séquentiellement de $k = 1$ à K – on écrit d'abord (n_1^E, i_1) puis (n_2^E, i_2) puis (n_3^E, i_3), etc. Les données du k-ième sous-vecteur sont réellement écrites si le nombre de bit restant est suffisant. Le nombre minimal de bits pour écrire (n_k^E, i_k) est 0 bit si $n'_k = 0$ et $5n'_k - 1$ si $n'_k \geq 2$.

L'écriture des données revient donc à :

1. Initialiser le tableau binaire *tab* à zéro, les pointeurs $pos_i = 1$ et $pos_n = B_{tot}$, et le nombre de bit restants $B' = B_{tot}$;
2. Pour $k = 1, \cdots, K$:
   (i) Calculer le nombre de bit minimal b pour écrire (n_k^E, i_k) : $b = 0$ si $n'_k = 0$ et $5n'_k - 1$ si $n'_k \geq 2$
   (ii) Si $0 < b \leq B'$,
      - écrire $n_k'^E$ bit par bit (sauf le bit stop du codage unaire) en mettant à 1 tous les éléments $tab(pos_n - n'_k + 2 \cdots pos_n)$ et décrémenter pos_n de $n'_k - 1$;
      - écrire i_k bit par bit dans $tab(pos_i \cdots pos_i + 4n'_k - 1)$ et incrémenter pos_i de $4n'_k$
   (iii) Mettre à jour le nombre de bit restants : $B' = B' - b$
   (iv) si $B' > 0$, écrire le bit stop du codage unaire (i.e. $tab(pos_n) = 0$) et décrémenter pos_n de 1.

Ce format de multiplexage a été généralisé au cas où les paramètres sont écrits dans plusieurs paquets binaires [38].

Quantification du gain TCX

Le gain g (optimisé) est re-calculé et quantifié après codage multi-débit. Sa valeur est en fait re-calculée afin de minimiser

$$\|\mathbf{x} - g\hat{\mathbf{x}}'\|^2, \tag{6.8}$$

où $\mathbf{x}$ est la cible (dans le temps) et $\hat{\mathbf{x}}'$ est la transformée de Fourier discrète inverse de $\hat{\mathbf{X}}'$. On trouve facilement :

$$g = \hat{\mathbf{x}}'\mathbf{x}^t / \|\hat{\mathbf{x}}'\|^2. \tag{6.9}$$

On cherche ainsi à se rapprocher de la forme d'onde $\mathbf{x}$. Le gain g obtenu à l'équation 6.9 est quantifié sur 7 bits suivant une échelle logarithmique.

Calcul et quantification du niveau de bruit $\hat{\sigma}_{inj}$

Le niveau de bruit (plat en fréquences) à injecter peut être simplement fixé à la valeur RMS des sous-vecteurs codés à 0 (à partir d'une fréquence f_{inj}). De cette façon, l'énergie moyenne des sous-vecteurs non-quantifiés est conservée. Le niveau de bruit est quantifié sur 3 bits suivant une échelle linéaire relativement à g.

Injection de bruit

Pour masquer les artefacts de codage, du bruit est injecté au décodage dans les sous-vecteurs reconstruits à zéro dans la bande 1600-6400 Hz. Cette injection de bruit est similaire à [39]. Elle est de plus justifiée dans le contexte du codage ACELP/TCX multi-mode pour rendre la commutation de modes inaudibles [6]. Le bruit de confort est géneré en fréquences sous la forme $cos(\theta) + jsin(\theta)$, avec une phase θ aléatoire. En pratique les parties réelles et imaginaires sont tirées d'une table. Le bruit de confort a donc un spectre d'amplitude plat (l'énergie est constante égale à 1) ; il est mis à l'échelle par $g \times \sigma_{inj}$. Le caractère aléatoire de la phase est simulé à partir d'un générateur à congruence linéaire [40, chap. 3].

6.3.4 Remarque : réduction de complexité

La complexité du codage algébrique détaillé ici peut être réduite de plusieurs manières :

- L'optimisation du gain TCX, g, peut être réduite à la seule initialisation de g à partir de l'énergie des sous-vecteurs ;
- Les indices i_k peuvent être calculés après avoir fixé la valeur du gain g – ce changement n'affecte pas la qualité du codage TCX ;
- Le niveau de bruit σ_{inj} peut être estimé pendant l'optimisation de g à partir de l'énergie des sous-vecteurs et des numéros de dictionnaires estimés.

6.4 Évaluation du codage algébrique de la cible TCX

On présente ici des résultats obtenus à l'aide du modèle AMR-WB+, plus précisément du TCX sans prédiction de pitch appliqué dans une bande de fréquences allant de 0 à 6400 Hz.

6.4.1 Caractéristiques réelles de la cible TCX

La distribution des sous-vecteurs $\mathbf{X}_k$ de dimension 8 a été estimée à l'aide d'une base de signaux audio d'une durée de 320 s (plus de 5 mn), constituée d'une grande majorité de signaux de musique et de quelques séquences de parole. Les résultats sont montrés à la figure 6.14.

On rappelle que le codage TCX du modèle AMR-WB+ est présenté brièvement à l'annexe 6.C. Plusieurs longueurs de trame sont définies : 20, 40 et 80 ms. Un fenêtrage avec recouvrement est appliqué avant FFT, si bien que la cible transformée $\mathbf{X}$ est de dimension 288 pour des trames de 20 ms, 576 pour des trames de 40 ms et 1152 pour des trames de 80 ms. Pour ne pas biaiser les statistiques, les trames de silence ont été supprimées. Au total, 15940 trames de 20 ms, 7306 trames de 40 ms et 3667 trames de 80 ms ont été utilisées.

Les statistiques ont été calculées en déterminant pour chaque cible transformée $\mathbf{X} = [\mathbf{X}_1\ \mathbf{X}_2\ \cdots\ \mathbf{X}_K]$ le vecteur

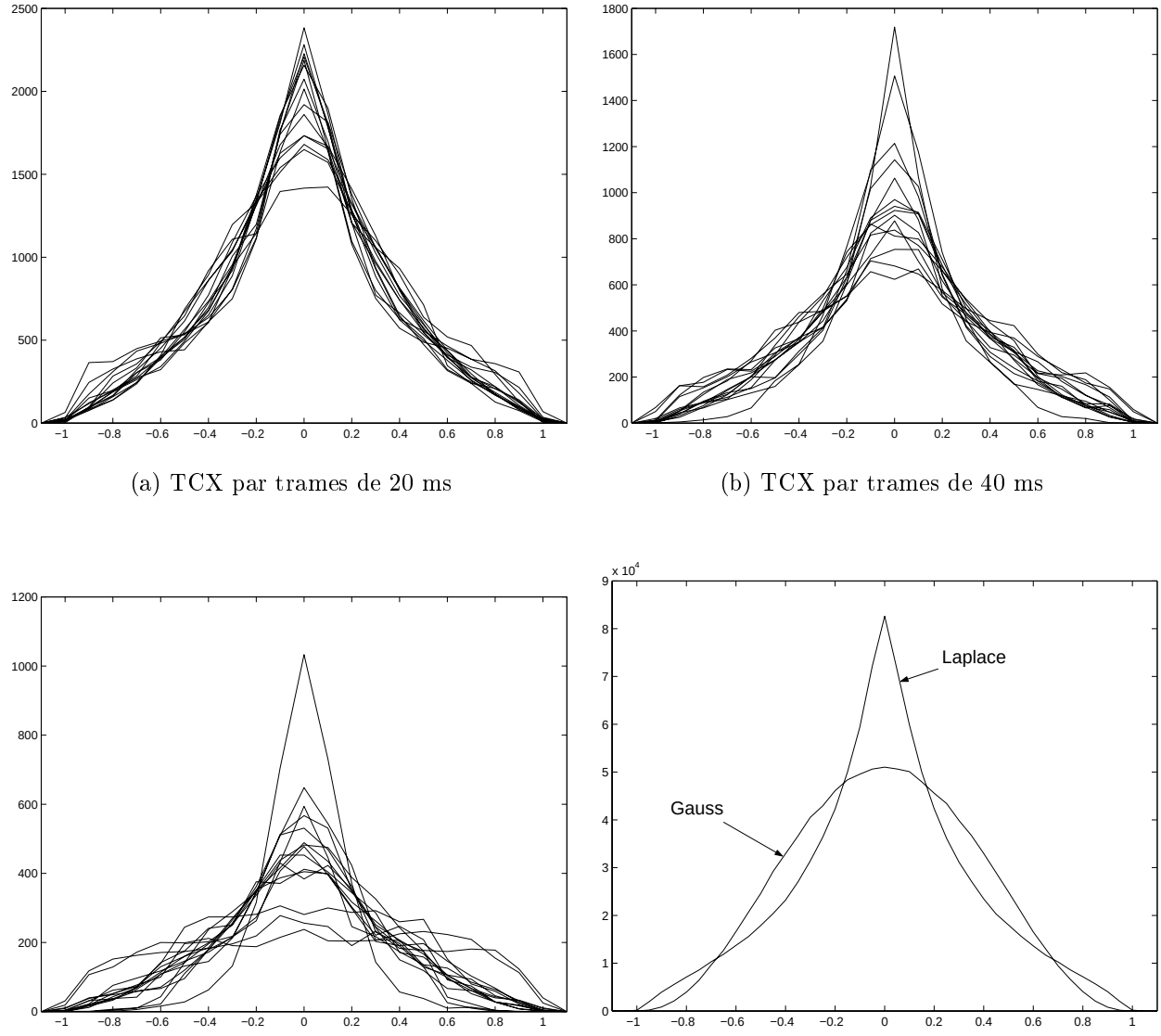
$$\mathbf{Y} = \left[\frac{1}{\|\mathbf{X}_1\|}\mathbf{X}_1\ \ \frac{1}{\|\mathbf{X}_2\|}\mathbf{X}_2\ \ \cdots\ \ \frac{1}{\|\mathbf{X}_K\|}\mathbf{X}_K\right] \tag{6.10}$$

et en calculant l'histogramme de Y_i sachant que $\mathbf{Y} = (Y_1, \cdots, Y_N)$. Ce calcul est raisonnable en modélisant la source $\mathbf{X}$ comme une source non-stationnaire obtenue en appliquant un facteur d'échelle aléatoire par sous-vecteurs à une source i.i.d. Les courbes des figures 6.14 (a), (b) et (c) représentent ainsi plusieurs histogrammes des composantes Y_i de $\mathbf{Y}$ superposées pour différents i relatifs à des coefficients de Fourier particuliers.

Pour comparaison la figure 6.14 (d) représente Y_i dans l'hypothèse où les sous-vecteurs de $\mathbf{Y}$ sont i.i.d., de moyenne nulle et de variance unité, et distribués selon la loi de Gauss ou la loi de Laplace.

Ces résultats montrent que les composantes des sous-vecteurs $\mathbf{X_k}$ de la cible TCX sont relativement i.i.d. et qu'elles sont plutôt laplaciennes. Cette conclusion est valide dans le cas du codage TCX par trames de 20 ms pour lequel les statistiques obtenues sont relativement fiables – dans les autres cas (40 et 80 ms) la quantité de données est insuffisante et les conclusions doivent être nuancées. Ces résultats contrastent avec les caractéristiques gaussiennes de la cible TCX mises en évidence dans le cas du codage TCX avec prédiction de pitch dans [3]. Ils justifient le choix des leaders absolus donnés au paragraphe 4.5.1 dans le tableau 4.3 – on peut vérifier que ces leaders absolus sont adaptés à une source non-gaussienne car ils ne correspondent pas une vraie troncature sphérique de RE_8.

On peut également prévoir à partir de ces résultats que pour le modèle TCX utilisé, le codage par troncature de Voronoï adaptative est moins performant que la quantification par extension de Voronoï. En effet, le codage par troncature de Voronoï adaptative utilise des dictionnaires Q_n quasi-sphériques adaptés uniquement à la source gaussienne, alors que la quantification par extension de Voronoï permet de s'adapter à une source non-gaussienne en optimisant de façon adéquate ses dictionnaires de base Q_2, Q_3 et Q_4 (par sélection statistique de leaders par exemple).



(a) TCX par trames de 20 ms

(b) TCX par trames de 40 ms

(c) TCX par trames de 80 ms

(d) sources de référence

Figure 6.14: Distribution des composantes de $\frac{1}{||\mathbf{X}_k||}\mathbf{X}_k$ pour des longueurs de trames de 20, 40 et 80 ms – en comparaison avec les sources gaussienne et laplacienne.

6.4.2 Performances

Pour évaluer le codage TCX indépendamment du choix des modes, le signal d'entrée est codé par le modèle AMR-WB+ en forçant un mode TCX unique (20, 40 ou 80 ms). La classification en boucle

fermée du codage ACELP/TCX multi-mode et la commutation de modes sont ainsi désactivées dans l'AMR-WB+. Le codage de la cible TCX est effectué tel que décrit à la section 6.3. Les résultats présentés ici se rapportent uniquement au codage du signal dans la bande basse (0–6400 Hz).

Allocation des bits

L'allocation des bits aux paramètres de codage TCX est détaillée aux tableaux 6.2 (a), (b) et (c). Celle-ci dépend de la longueur des trames (20, 40 ou 80 ms) et du débit de codage (de 12.8 à 24 kbit/s) du signal décimé à 12.8 kHz.

Les coefficients de prédiction linéaire sont codés sous forme de paires d'imittances spectrales (en anglais ISF pour Imittance Spectral Frequencies) avec le quantificateur prédictif de 46 bits du codage AMR-WB. De plus, le gain TCX g est codé sur 7 bits (sur une échelle logarithmique absolue), tandis que le niveau de bruit σ_{inj} est ici représenté sur 3 bits (sur une échelle linéaire relative, fonction de g). Ces paramètres ne sont codés qu'une fois par trame de 20, 40 ou 80 ms ; les coefficients LPC sont interpolés par sous-trames de 5 ms.

De cette façon le débit relatif alloué au codage de la cible augmente avec la longueur de trame. Par exemple, à 12.8 kbit/s, la cible TCX (normalisée) est codée à un débit de 0.69 bit par échantillon avec une longueur de trame de 20 ms, contre 0.78 et 0.83 pour des longueurs de trame de respectivement 40 et 80 ms. Avec cette allocation, le codage TCX par trames de 40 et 80 ms est donc adapté à des signaux dont le contenu spectral est relativement stationnaire sur respectivement 40 et 80 ms.

Performances à différents débits et longueurs de trame

Les performances du codage TCX sont présentées au tableau 6.3, dans le cas où la cible TCX est codée à l'aide de la quantification multi-débit dans RE_8 par extension de Voronoï. Les performances sont évaluées de façon objective en terme de rapport signal-sur-bruit (RSB) segmentaire dans le

TABLEAU 6.2: Allocations des bits aux paramètres du modèle TCX pour différents débits de codage – les nombres entre paranthèses correspondent au débit par dimension alloué au codage de la cible TCX.

(a) codage TCX par trames de 20 ms

Paramètre du modèle TCX	Débit (kbit/s)					
	12.8	14.4	16.0	18.4	20.0	24.0
Coefficients LPC (ISF)	46					
Gain TCX g	7					
Niveau de bruit σ_{inj}	3					
Cible normalisée $\mathbf{X}'$ de dimension 288	198 (0.69)	230 (0.80)	262 (0.91)	310 (1.08)	342 (1.19)	422 (1.47)

(b) codage TCX par trames de 40 ms

Paramètre du modèle TCX	Débit (kbit/s)					
	12.8	14.4	16.0	18.4	20.0	24.0
Coefficients LPC (ISF)	46					
Gain TCX g	7					
Niveau de bruit σ_{inj}	3					
Cible normalisée $\mathbf{X}'$ de dimension 576	452 (0.78)	516 (0.90)	580 (1.01)	676 (1.17)	740 (1.28)	900 (1.56)

(c) codage TCX par trames de 80 ms

Paramètre du modèle TCX	Débit (kbit/s)					
	12.8	14.4	16.0	18.4	20.0	24.0
Coefficients LPC (ISF)	46					
Gain TCX g	7					
Niveau de bruit σ_{inj}	3					
Cible normalisée $\mathbf{X}'$ de dimension 1152	960 (0.83)	1088 (0.94)	1216 (1.06)	1408 (1.22)	1536 (1.33)	1856 (1.61)

domaine de la cible (temporelle) :

$$\mathrm{RSB}_{seg_w} = \frac{1}{M}\sum_{k=1}^{M}\mathrm{RSB}(i), \tag{6.11}$$

où M est le nombre de sous-trames et $\mathrm{RSB}(i)$ est le rapport signal-sur-bruit dans la sous-trame i :

$$\mathrm{RSB}(i) = 10\log_{10}\frac{\sum_n x^2(n)}{\sum_n (x(n)-\hat{x}(n))^2} \tag{6.12}$$

et $x(n)$ et $\hat{x}(n)$ correspondent respectivement à la cible (temporelle) et sa version quantifiée. La longueur des sous-trames est ici fixée à 5 ms (64 échantillons).

Plusieurs signaux en bande élargie sont traités par le modèle AMR-WB+ : de la parole (voix d'homme et de femme) et des instruments de musique (orgue et harpe). On ne retient ici que le cas de la voix d'homme et de l'orgue, car ils sont représentatifs de la base de signaux audio utilisée. Les résultats objectifs du tableau 6.3 sont représentés sous forme graphique à la figure 6.15.

TABLEAU 6.3: Rapport signal-sur-bruit segmentaire dans le domaine de la cible (temporelle) RSB_{seg_w} (en dB) pour différents signaux avec un codage de la cible TCX à l'aide de la quantification multi-débit dans RE_8 par extension de Voronoï.

Signal	Mode	Débit (kbit/s)					
		12.8	14.4	16.0	18.4	20.0	24.0
parole (voix masculine)	TCX 20 ms	6.65	7.14	7.69	8.48	9.00	10.06
	TCX 40 ms	5.84	6.34	6.85	7.58	8.06	9.10
	TCX 80 ms	3.32	3.84	4.32	4.96	5.41	6.39
	ACELP/TCX	8.74	9.36	9.96	10.82	11.27	12.40
orgue	TCX 20 ms	8.45	9.32	10.11	11.27	11.99	13.34
	TCX 40 ms	11.24	12.13	12.93	14.08	14.81	16.15
	TCX 80 ms	13.37	14.27	15.09	16.18	16.85	18.11
	ACELP/TCX	13.70	14.60	15.42	16.53	17.19	18.46
parole (voix féminine)	TCX 20 ms	7.50	8.05	8.64	9.47	10.02	11.17
	TCX 40 ms	6.78	7.26	7.78	8.59	9.18	10.26
	TCX 80 ms	5.12	5.60	6.06	6.80	7.33	8.32
harpe	TCX 20 ms	9.08	9.94	10.78	11.90	12.59	13.83
	TCX 40 ms	11.34	12.14	12.87	13.90	14.51	15.69
	TCX 80 ms	11.89	12.62	13.29	14.26	14.85	16.02

On peut tout d'abord vérifier que pour une longueur de trame fixée (20, 40 ou 80 ms), les performances du codage TCX augmentent de façon relativement linéaire avec le débit. Ces résultats montrent l'intérêt du codage algébrique TCX décrit ici et sont la conséquence de l'application du principe du remplissage inverse des eaux. En effet, ce principe garantit en effet sous certaines conditions des performances qui suivent la borne débit-distorsion (linéaire en terme de rapport signal-sur-bruit).

Par ailleurs, à un débit fixé, on peut valider l'intérêt de ne coder qu'une seule fois par trame les coefficients de prédiction afin de coder au mieux les signaux stationnaires sur 20, 40 ou 80 ms.

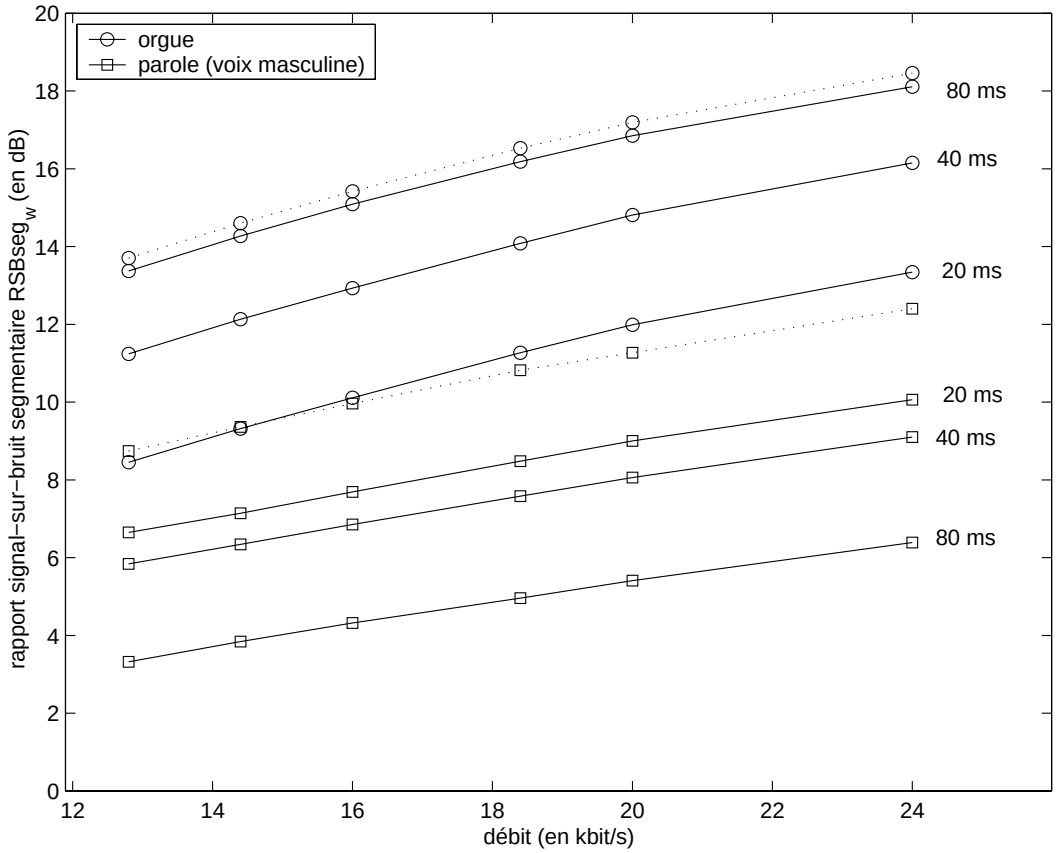


Figure 6.15: Qualité objective du codage TCX avec des longueurs de 20, 40 et 80 ms pour des signaux d'orgue et de parole en bande élargie (décimés à 12.8 kHz) – les courbes en pointillés montrent la performance obtenue avec un codage ACELP/TCX multi-mode (alternant entre ACELP ou TCX à 20, 40 ou 80 ms).

En effet, pour le signal d'orgue, qui est un signal relativement stationnaire sur de longs segments temporels, la performance augmente de façon significative avec la longueur de trame. Le gain de performance peut être attribué dans ce cas au gain de codage par transformée et surtout à l'augmentation du débit relatif alloué au codage de la cible TCX normalisée.

Par contre on peut constater que le codage TCX tel qu'employé dans l'AMR-WB+ ne peut coder à lui seul la parole ; la performance se dégrade quand la longueur des trames augmente. Le codage

TCX employé n'est en effet adapté qu'à des sons stationnaires (comme des segments d'orgue ou de harpe) et provoque pour les signaux tels que la parole un étalement temporel du bruit de codage et une distorsion spectrale due à l'interpolation des filtres LPC sur de longs segments. De plus, la prédiction de pitch n'est pas utilisée afin de coder la cible TCX sur de longues trames, comme dans [20].

A des fins de comparaison, les courbes en pointillés à la figure 6.15 montrent les performances atteintes en autorisant la commutation de modes entre ACELP et TCX à 20, 40 et 80 ms. On peut observer que la performance pour l'orgue est en grande partie due au codage TCX 80 ms ; à l'opposé, pour la parole (voix d'homme), la qualité (objective) est due essentiellement au codage ACELP du modèle AMWR-WB+.

Observations des signaux décodés

Les performances objectives mesurées pour les signaux de parole peuvent être expliquées à l'aide de la figure 6.16. On met ainsi en évidence les défauts classiques du codage par transformée pour la parole : pré-écho, attaques mal définies, etc. Puisqu'en codage TCX par trame de 80 ms, les coefficients LPC sont codés une fois par trame, l'enveloppe spectrale est mal représentée. On peut observer ce défaut en examinant les réponses impulsionnelles localisées sur chaque impulsion glottale.

Les bonnes performances du codage TCX pour l'orgue peuvent également être expliquées à l'aide de l'exemple de la sous-section 6.3.2. En augmentant le débit de codage, on abaisse le seuil λ, défini à la figure 6.11 et au paragraphe 6.3.2, qui distingue les sous-vecteurs codés de ceux qui sont mis à zéro. D'un point de vue qualitatif, ce codage équivaut à abaisser l'enveloppe spectrale de bruit de codage, tel qu'illustré à la figure 6.17.

Comparaison des performances de la quantification multi-débit dans RE_8

Les performances du codage TCX sont présentées au tableau 6.4, cette fois-ci dans le cas où la cible TCX est codée à l'aide de la quantification multi-débit dans RE_8 par codage de Voronoï dans

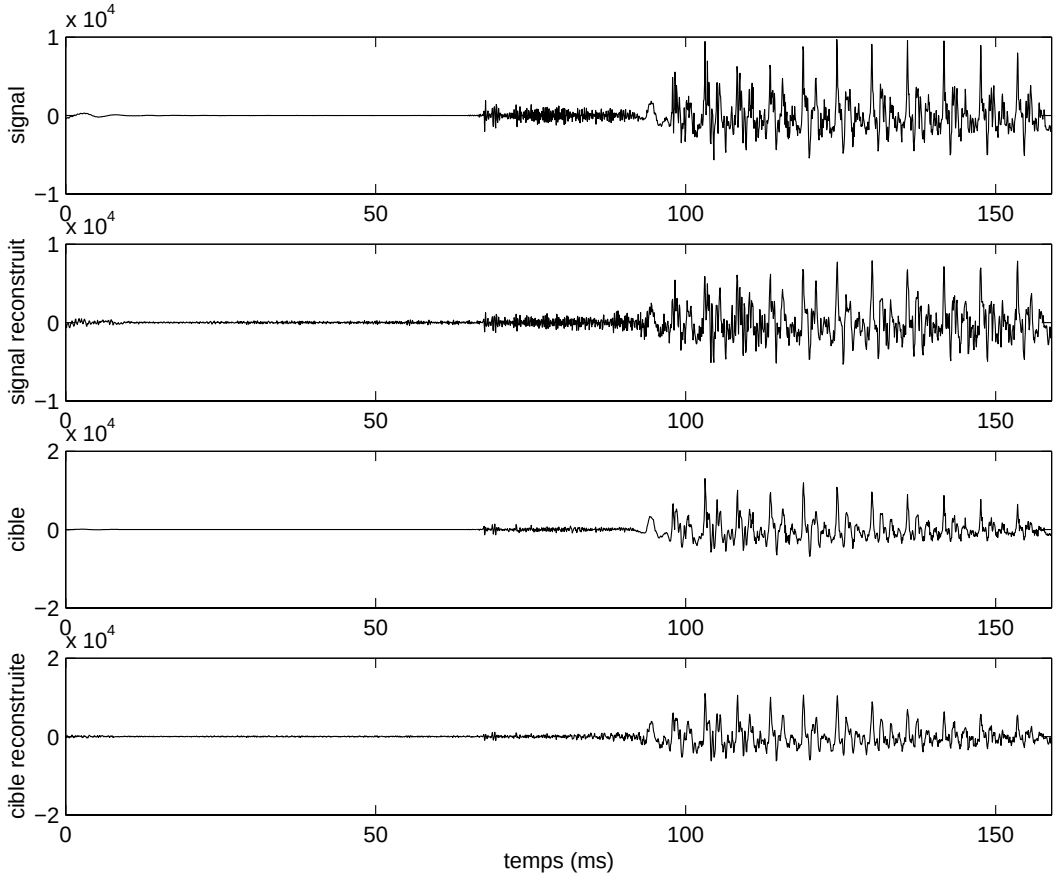


Figure 6.16: Exemple de codage TCX par trames de 80 ms du signal de parole (voix masculine) : de haut en bas, signal de parole original (décimé à 12.8 kHz), signal reconstruit, cible originale et cible reconstruite.

$RE_8/2E_8^\star$. A des fins de comparaison, les résultats des tableaux 6.3 et 6.4 sont représentés sous forme graphique à la figure 6.18 dans le cas de l'orgue uniquement.

On rappelle qu'entre la quantification multi-débit dans RE_8 par extension de Voronoï ou par codage de Voronoï dans $RE_8/2E_8^\star$, les dictionnaires Q_n et l'indexation multi-débit dans RE_8 diffèrent. Le réseau de points, le codage des numéros de dictionnaire et le principe de multiplexage des paramètres sont identiques.

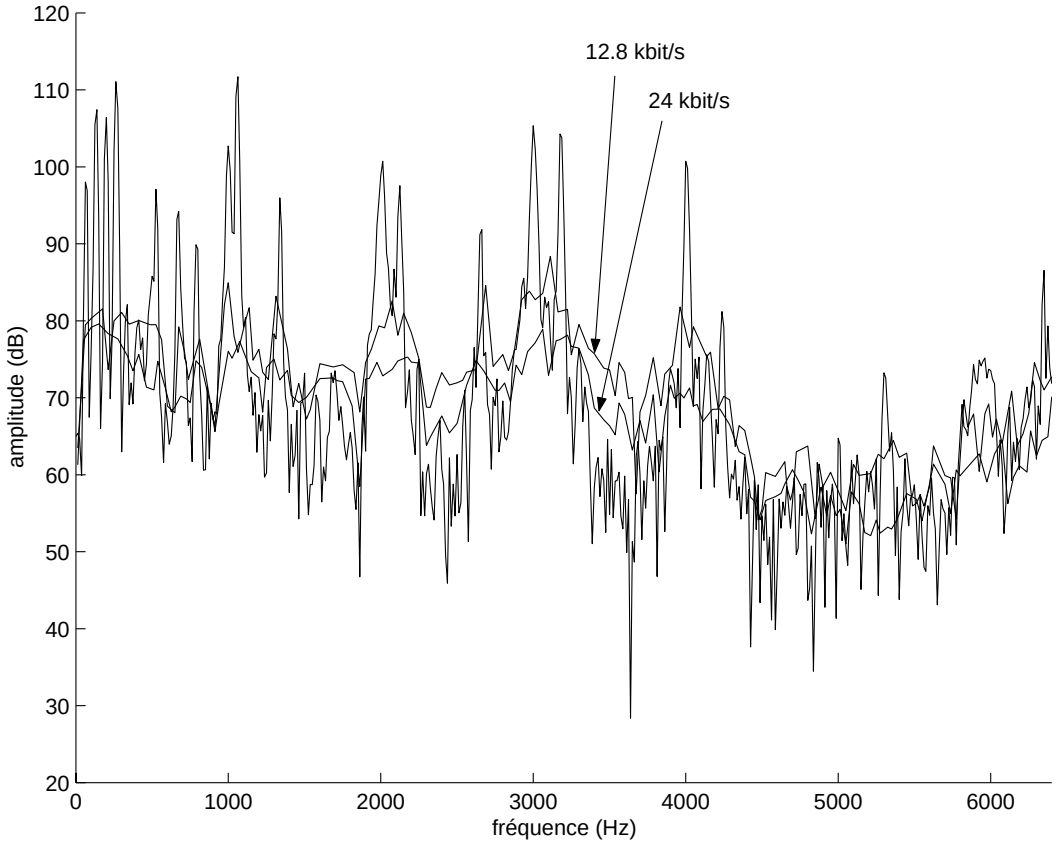


Figure 6.17: Exemple de codage TCX par trames de 80 ms du signal d'orgue à 12.8 et 24 kbit/s. On distingue le spectre du signal, et les enveloppes spectrales des bruits de codage à 12.8 et 24 kbit/s.

Ces résultats peuvent être facilement interprétés en se reportant à la figure 5.17. Dans la gamme des débits utilisés ici (entre 0.69 et 1.61 bit par dimension), le codage dans $RE_8/2E_8^\star$ est en effet peu efficace. De plus, les statistiques de la sous-section 6.4.1 montrent que la cible a une distribution plus laplacienne que gaussienne (pour les signaux de musique tout au moins). Or, le codage de Voronoï est plutôt adapté à une source gaussienne. Le codage de Voronoï dans $RE_8/2E_8^\star$ est donc moins performant ici que la quantification multi-débit par extension de Voronoï.

On peut également observer que l'écart de performance augmente lorsque la longueur des trames diminue. Ce comportement peut être expliqué par le fait que, pour des trames courtes, l'énergie est

TABLEAU 6.4: Rapport signal-sur-bruit segmentaire dans le domaine de la cible (temporelle) RSB_{seg_w} (en dB) pour différents signaux avec un codage de la cible TCX à l'aide du codage de Voronoï dans $RE_8/2E_8^\star$.

Signal	Mode	Type de codage	Débit (kbit/s)					
			12.8	14.4	16.0	18.4	20.0	24.0
parole (voix masculine)	TCX 20 ms	$RE_8/2E_8^\star$	5.94	6.43	6.83	7.46	7.90	8.81
		hybride	6.53	7.02	7.55	8.34	8.85	9.90
	TCX 40 ms	$RE_8/2E_8^\star$	5.14	5.57	5.98	6.61	7.05	7.86
		hybride	5.75	6.24	6.74	7.46	7.95	8.91
	TCX 80 ms	$RE_8/2E_8^\star$	2.74	3.11	3.50	4.13	4.57	5.31
		hybride	3.29	3.78	4.28	4.91	5.34	6.31
orgue	TCX 20 ms	$RE_8/2E_8^\star$	6.88	7.70	8.45	9.55	10.21	11.59
		hybride	7.96	8.76	9.51	10.55	11.25	12.55
	TCX 40 ms	$RE_8/2E_8^\star$	9.71	10.67	11.58	12.76	13.53	14.90
		hybride	10.60	11.49	12.30	13.47	14.18	15.57
	TCX 80 ms	$RE_8/2E_8^\star$	12.38	13.30	14.13	15.22	15.90	17.12
		hybride	12.99	13.89	14.72	15.81	16.51	17.79

répartie plus uniformément entre coefficients transformés et les dictionnaires Q_n de bas débits (en particulier Q_2) influencent plus la performance.

6.4.3 Statistiques pertinentes

Pour analyser plus en profondeur la performance du codage TCX algébrique, on montre à la figure 6.19 la fréquence des numéros de dictionnaire n. Ces statistiques ont été mesurées à l'aide du codage TCX par trames de 20 ms uniquement, avec une quantification multi-débit dans RE_8 par extension de Voronoï. La base de données audio utilisée est décrite à la sous-section 6.4.1.

Ces résultats montrent que le codage unaire utilisé pour coder la cible TCX est sous-optimal. On pourrait ainsi envisager un codage séparé des numéros $n = 0$ (par plages ou autre) – les numéros $n \geq 2$ étant codés différemment.

Le tableau 6.5 montre les leaders absolus dans RE_8 les plus fréquents. Les statistiques sous-jacentes sont obtenues à l'aide de $\tilde{\mathbf{X}}'$ pour la base de signaux audio décrites au paragraphe 6.4.1. Pour pouvoir comparer différents leaders, leur fréquence (ou probabilité estimée d'occurence) est

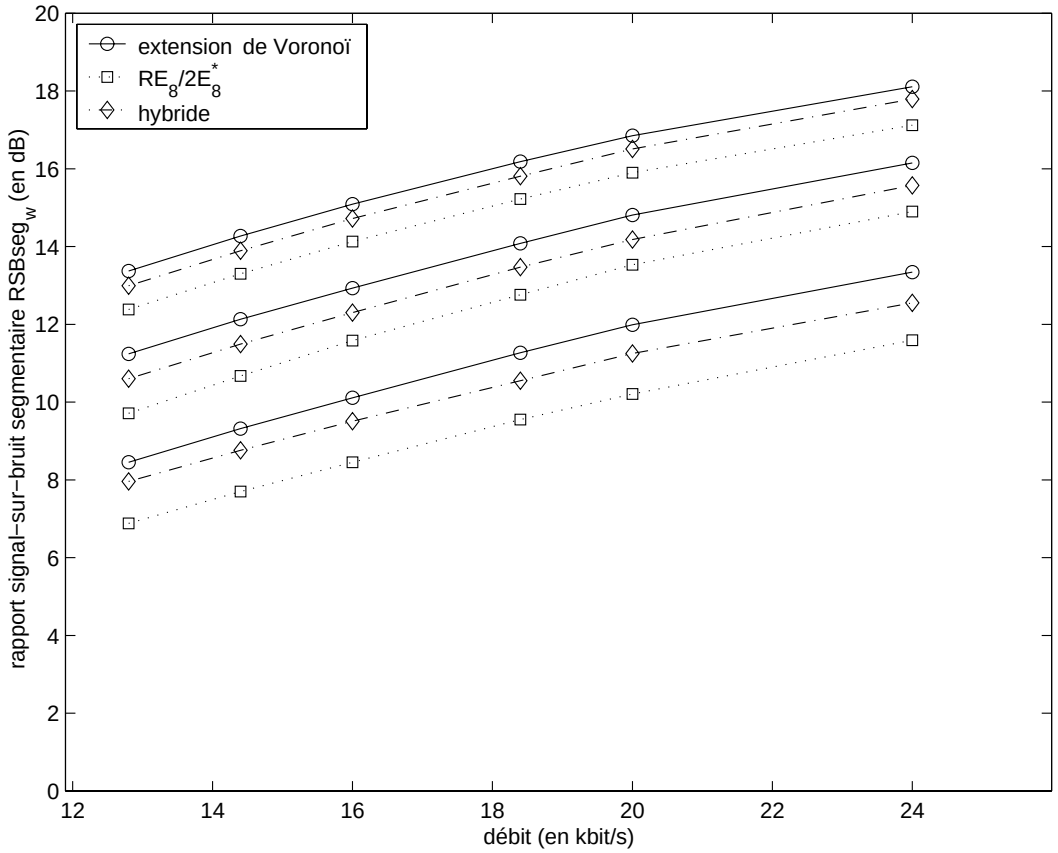
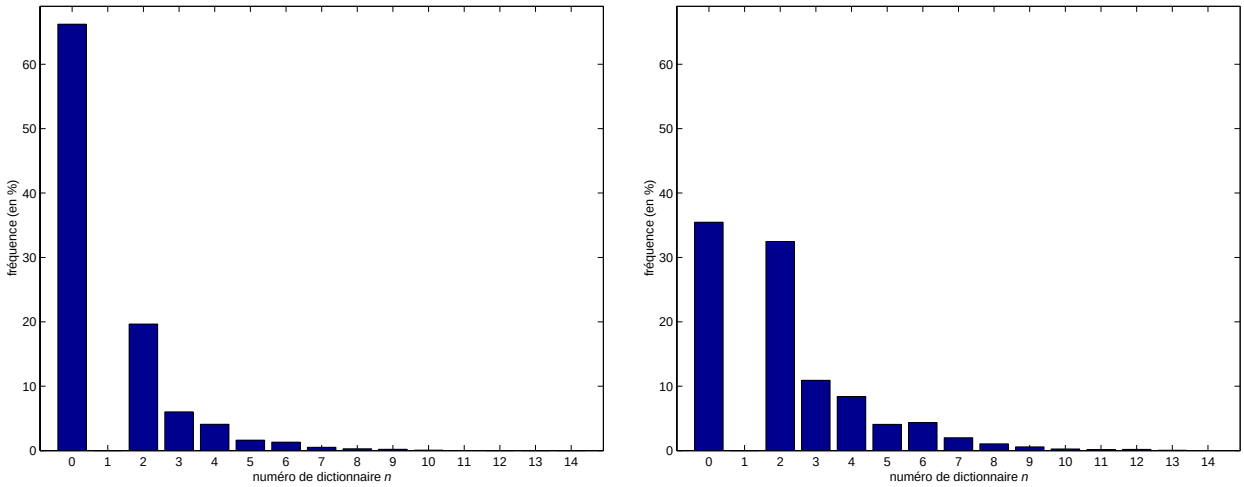


Figure 6.18: Qualité objective du codage TCX avec des longueurs de 20, 40 et 80 ms pour le signal d'orgue en bande élargie (décimés à 12.8 kHz) et différents type de codage multi-débit dans RE_8.

normalisée par leur nombre de permutations signées dans RE_8. Cet ordre change très peu avec le débit (en allant de 12.8 à 24 kbit/s).

Ce tableau permet de vérifier que les sous-vecteurs de la cible TCX sont quasi-laplaciens car les leaders les plus probables se trouvent sur des orbites pyramidales de rayon de plus en plus élevé.



(a) 12.8 kbit/s (b) 24 kbit/s

Figure 6.19: Fréquence des numéros de dictionnaire n pour une quantification multi-débit dans RE_8 par extension de Voronoï.

6.5 Conclusions

Ce chapitre a présenté une nouvelle technique de codage de la cible TCX. Celle-ci applique une quantification multi-débit par réseaux de points de type gain-forme, avec une allocation des bits adaptative implicite contrôlée suivant le principe du remplissage des eaux (*reverse waterfilling*). Ce principe donne un fondement théorique rigoureux au codage TCX.

Le codage algébrique de la cible, tel qu'introduit dans [3], est rendu plus flexible. Celui-ci repose sur une recherche d'un gain unique (le gain TCX) dans un domaine logarithmique. Il peut s'adapter à des longueurs de trames et des débits différents, tout en représentant la variabilité spectrale ; sa complexité est en outre faible car la quantification opère sans saturation complexe, à la différence de [3].

Un système de codage à partir du réseau RE_8 a été décrit pour illustrer ce principe de codage. Plusieurs aspects connexes n'ont pas été présentés par souci de concision. En particulier, le multiplexage en plusieurs paquets des paramètres de codage, ainsi que la correction des pertes de trame

TABLEAU 6.5: Leaders absolus de RE_8 triés par ordre décroissant de fréquences – pour un codage TCX à 20 ms et 12.8 kbit/s.

Leader absolu dans RE_8 $\mathbf{y} = (y_1, \cdots, y_8)$								Norme $\sum_i \lvert y_i \rvert$
0	0	0	0	0	0	0	0	0
2	2	0	0	0	0	0	0	4
4	0	0	0	0	0	0	0	4
1	1	1	1	1	1	1	1	8
8	0	0	0	0	0	0	0	8
3	1	1	1	1	1	1	1	8
4	4	0	0	0	0	0	0	8
2	2	2	2	0	0	0	0	8
6	2	0	0	0	0	0	0	8
4	2	2	0	0	0	0	0	8
6	6	0	0	0	0	0	0	12
3	3	1	1	1	1	1	1	12
12	0	0	0	0	0	0	0	12
5	1	1	1	1	1	1	1	12
8	4	0	0	0	0	0	0	12
10	2	0	0	0	0	0	0	12
2	2	2	2	2	2	0	0	12
8	2	2	0	0	0	0	0	12
6	4	2	0	0	0	0	0	12
4	2	2	2	2	0	0	0	12
			⋮					⋮

ont été passés sous silence.

Ce travail pourrait être étendu de plusieurs façons :

- L'allocation des bits repose ici sur le principe du remplissage inverse des eaux. Cette procédure pourrait être remplacée par un algorithme d'allocation des bits optimal comme, par exemple, l'algorithmes de Shoham et Gersho [41] ou celui de Riskin [42].

- La cible est codée par transformée discrète de Fourier, ce qui permet en particulier d'exploiter les propriétés fréquentielles de la perception auditive ; néanmoins, aucun modèle perceptuel n'est exploité ici car on suppose que le filtre perceptuel inverse $W(z)^{-1}$ réalise un masquage suffisant. Cette hypothèse peut néamoins être remise en cause à bas débit comme en té-

moignent les ajustements au modèle TCX décrits à l'annexe 6.C Par ailleurs à haut débit, rien ne garantit que les bits sont alloués aux bons sous-vecteurs. On pourrait donc utiliser un modèle perceptuel permettant de calculer des facteurs d'échelle par bande ou une correction de l'enveloppe spectrale du filtre perceptuel.

- Le codage s'est appuyé ici sur le réseau RE_8 avec un codage unaire des numéros de dictionnaire. D'autres réseaux pourraient être testés – en particulier à dimension plus élevée. Un codage plus évolué des numéros de dictionnaire pourrait être conçu. Des aspects comme l'optimisation de la granularité en débit des dictionnaires Q_n peuvent également être considérés.

Annexe 6.A : Complexité du codage CELP à débit élevé

Les vecteurs sont ici des vecteurs colonnes.

Principe du codage CELP de la parole

Le modèle CELP est un modèle de codage hybride dans le sens où il consiste à coder une forme d'onde $s(n)$ à partir d'un modèle paramétrique du signal [43]. La synthèse $\hat{s}(n)$ est produite par le modèle source-filtre linéaire prédictif décrit à la figure 6.20. Ce modèle comprend 3 parties : la prédiction linéaire (filtre inverse $1/A(z)$), la prédiction de pitch (dictionnaire adaptatif $p(n)$ et gain de pitch g_p) et le codage de l'innovation ($c(n)$ et g_c) L'excitation $\hat{r}(n)$ est codée en boucle fermée suivant le principe de l'analyse par synthèse [44]. Les paramètres associés sont donc optimisés en minimisant l'énergie de l'erreur pondérée $e_w(n)$ c'est-à-dire l'erreur $e(n) = s(n) - \hat{s}(n)$ filtrée par le filtre perceptuel $W(z)$.

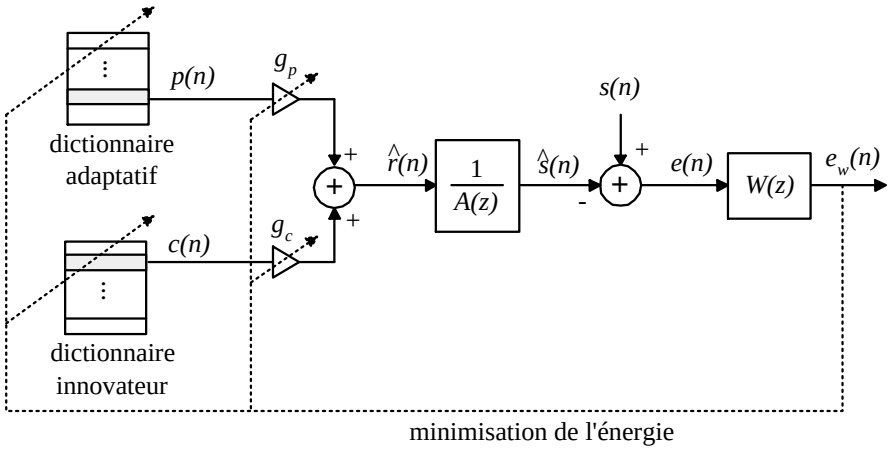


Figure 6.20: Modèle paramétrique CELP avec analyse par synthèse.

Le signal de parole $s(n)$ est codé par blocs (ou trames) de l'ordre de 10 à 30 ms pour que l'hypothèse de stationnarité de la prédiction linéaire s'applique. Pour avoir une complexité raisonnable, les paramètres sont codés séquentiellement : les coefficients du filtre $1/A(z)$ sont d'abord quantifiés

au niveau de la trame, puis dans chaque sous-trame la prédiction de pitch est appliquée avant le codage de l'innovation. La longueur de la sous-trame est typiquement de l'ordre de 5–7 ms. Cette recherche par sous-trame est décrite à la figure 6.1 (a) dans le cas où $W(z) = \hat{A}(z)/\hat{A}(z/\gamma)$.

L'élimination du "ringing", c'est-à-dire la réponse libre $s_{w0}(n)$ du filtre $1/\hat{A}(z/\gamma)$, permet de simplifier le traitement en bloc , en remplaçant le filtrage par une multiplication vectorielle par une matrice de convolution.

La prédiction linéaire (court-terme) et la prédiction de pitch sont revues dans [44]. Le problème du codage CELP se ramène essentiellement à la recherche dans le dictionnaire innovateur, donc à minimiser dans chaque sous-trame

$$E = \arg\min_{g_c, \mathbf{c_k}} \|\mathbf{x} - g_c H \mathbf{c_k}\|^2 \tag{6.13}$$

où $\mathbf{x}$ est le vecteur cible, g_c est le gain du dictionnaire innovateur, H la matrice de réponse impulsionnelle du filtre $W(z)/\hat{A}(z)$ à mémoire nulle (H est triangulaire inférieure) et $\mathbf{c_k}$ est le k-ième mot du dictionnaire innovateur. Pour $\mathbf{c}_k$ fixé, le gain g_c optimal (au sens de E) est donné par

$$g_c = \frac{\mathbf{x}^t(H\mathbf{c_k})}{\mathbf{c_k}^t \Phi \mathbf{c_k}}, \tag{6.14}$$

où $\Phi = H^t H$ est la matrice d'autocorrélation de H ayant une structure de Toeplitz. Pour cette valeur optimale de g_c, la minimisation de E revient à trouver

$$\arg\max_{\mathbf{c_k}} \frac{(\mathbf{x}^t(H\mathbf{c_k}))^2}{\mathbf{c_k}\Phi\mathbf{c_k}^t}. \tag{6.15}$$

Le calcul du filtrage $H\mathbf{c_k}$ pour chaque c_k est coûteux. Il peut être évité car $\mathbf{x}^t(H\mathbf{c_k}) = \mathbf{d}^t c_k$ avec $\mathbf{d} = H^t\mathbf{x}$. En pré-calculant $\mathbf{d}$ et Φ, la recherche s'effectue directement dans le domaine de $\mathbf{c_k}$ sans filtrage [45, 46] :

$$\arg\max_{\mathbf{c_k}} \frac{(\mathbf{d}^t\mathbf{c_k})^2}{\mathbf{c_k}\Phi\mathbf{c_k}^t}. \tag{6.16}$$

Le calcul de $\mathbf{d} = H^t\mathbf{x}$ est appelé *backward filtering* dans [45] car $H^t\mathbf{x}$ correspond à un filtrage avec repliement temporel par rapport à $H\mathbf{c_k}$.

Dictionnaire ACELP

En codage ACELP, le dictionnaire innovateur est constitué d'impulsions d'amplitude ± 1 disposées dans des pistes entrelacées (en anglais *tracks*). Ce type de dictionnaires est appelé ISPP (*Interleaved Single-Pulse Permutations*) dans [47].

Cette contrainte simplifie énormément la recherche dans le dictionnaire innovateur et se justifie par le fait que les impulsions permettent de localiser dans le temps des impulsions glottales dans les segments voisés et les attaques.

Les numérateur et dénominateur de l'équation 6.16 sont donnés au tableau 6.5 pour 1, 2 et 3 impulsion(s).

TABLEAU 6.6: Numérateur et dénominateur du critère de recherche dans le dictionnaire innovateur pour un dictionnaire à impulsions (amplitude ± 1).

Nombre d'impulsions d'amplitude ± 1	Position(s) dans $[1, N]$	Numérateur $(\mathbf{d}^t\mathbf{c_k})^2$	Dénominateur $\mathbf{c_k}\Phi\mathbf{c_k}^t$
1	m_0	$d_{m_0}^2$	$\Phi(m_0, m_0)$
2	m_0, m_1	$(\pm d_{m_0} \pm d_{m_1})^2$	$\Phi(m_0, m_0)+$ $\Phi(m_1, m_1) \pm 2\Phi(m_0, m_1)$
3	m_0, m_1, m_2	$(\pm d_{m_0} \pm d_{m_1} \pm d_{m_2})^2$	$\Phi(m_0, m_0)+$ $\Phi(m_1, m_1) \pm 2\Phi(m_0, m_1)+$ $\Phi(m_2, m_2) \pm 2\Phi(m_0, m_2) \pm 2\Phi(m_1, m_2)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$

D'une manière générale, pour N impulsions de signes $s(i)$ et de position $p(i)$ dans la trame, on a :

$$\mathbf{d}^t\mathbf{c_k} = \sum_{i=1}^{N} s(i)d(p(i)) \tag{6.17}$$

et

$$\mathbf{c_k}\Phi\mathbf{c_k}^t = \sum_{i=1}^{N} s^2(i)\Phi(p(i), p(i)) + \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} s(i)s(j)\Phi(p(i), p(j)) \tag{6.18}$$

Plus le nombre d'impulsions augmente, plus le débit est élevé mais surtout plus la complexité de la recherche dans le dictionnaire innovateur devient problématique.

Pour optimiser le codage des impulsions et simplifier la recherche, en codage ACELP les positions sont entrelacées par pistes. Plusieurs stratégies de recherche par piste ont été développées dont :

- Recherche par boucles imbriquées avec seuillage (G.729) [48]
- Recherche en profondeur d'abord (G.729A) [49]
- Recherche partielle de type "profondeur d'abord" de plusieurs impulsions (simultanément) (G.729E,EFR) [50]

L'indexation des impulsions est revue et modifiée dans [36, 51].

Annexe 6.B : Allocation optimale de ressources en codage de source et de canal dans le cas gaussien

On rappelle ici des résultats classiques de théorie de l'information concernant, dans le cas *gaussien*, l'allocation optimale des bits pour l'erreur quadratique (en codage de source) et l'allocation optimale de puissances (en modulation). Ces résultats illustrent la dualité bien connue qui existe entre codage de source et codage de canal.

Allocation des puissances à des canaux gaussiens parallèles

Pour L canaux parallèles perturbés par un bruit blanc gaussien additif ($AWGN$), où le canal i est de largeur de bande B_i, la capacité totale est donnée par

$$C(P) = \sum_{i=1}^{L} B_i \log_2 \left(1 + \frac{\sigma_{X_i}^2}{\sigma_{N_i}^2}\right) \tag{6.19}$$

en bit/s, quand $\sigma_{X_i}^2$ est la puissance allouée au canal i et $\sigma_{N_i}^2$ est la puissance de bruit dans ce canal. Pour une contrainte de puissance totale

$$\sum_{i=1}^{L} \sigma_{X_i}^2 \leq P, \tag{6.20}$$

la capacité totale $C(P)$ est maximisée quand la puissance totale est distribuée suivant le principe du remplissage des eaux (*water-filling* ou *water-pouring*) [52, 5]. Un exemple est présenté à la figure 6.21, où φ désigne le "niveau d'eau".

La puissance allouée à chaque canal est donc $\sigma_{X_i}^2 = max(\varphi - \sigma_{N_i}^2, 0)$. L'allocation des puissances (*power control*) optimale est donc réalisée en cherchant la valeur de φ maximisant $C(P)$ sous la contrainte de puissance totale. Ce principe est couramment utilisé en communications numériques mono- ou multi-utilisateurs, par exemple en modulation CDMA ou multi-porteuse (OFDM, DMT) [53].

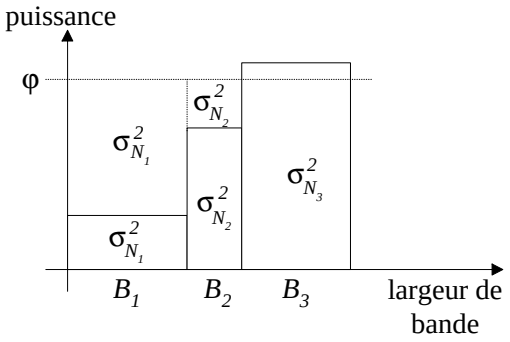


Figure 6.21: Allocation des puissances suivant le principe du remplissage des eaux.

Allocation des bits à des variables aléatoires gaussiennes

Pour K variables aléatoires gaussiennes de moyenne nulle et de variance $\sigma^2_{X_k}$, le débit total est donné par

$$R(D) = \sum_{k=1}^{K} \frac{1}{2} \log_2 \frac{\sigma^2_{X_k}}{D_k} \tag{6.21}$$

où D_k est la distorsion de la source k , définie comme l'*erreur quadratique* moyenne. Pour une contrainte de distorsion totale

$$\sum_{k=1}^{L} D_k \leq D, \tag{6.22}$$

le débit total est minimisé quand la distorsion totale est distribuée suivant le principe du remplissage inverse des eaux (*reverse water-filling*) [5] . Ce principe a été introduit par Kolmogorov dans [54] pour des sources continues. La distorsion D_k est donnée par $D_k = min(\lambda, \sigma^2_{X_k})$, où λ est le "niveau d'eau". Un exemple est montré à la figure 6.22. Si la puissance de la source k dépasse le seuil λ, la distorsion est fixée à λ ; autrement, aucun bit n'est alloué à cette source si bien que la distorsion est équivalente à la variance de la source $\sigma^2_{X_k}$.

La fonction $R(D)$ peut donc être déterminée point par point, en fixant d'abord λ puis en évaluant D et R pour cette valeur de λ ; une variation continue adéquate de λ conduit à la courbe débit-distorsion $R(D)$. La fonction $R(D)$ étant bijective, on obtient également en mesurant D et R pour chaque λ la fonction $D(R)$.

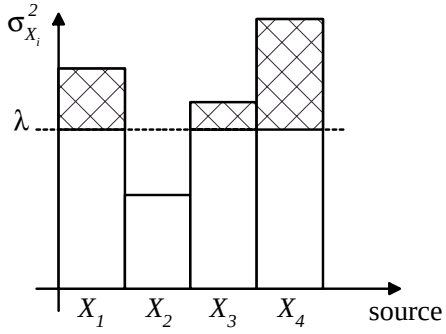


Figure 6.22: Allocation des bits suivant le principe du remplissage inverse des eaux.

En pratique, on cherche généralement à minimiser la distorsion totale

$$D(R) = \sum_{i=k}^{K} \sigma_{X_k}^2 2^{-2R_k}, \tag{6.23}$$

avec une contrainte de débit total

$$\sum_{i=1}^{K} R_k \leq R. \tag{6.24}$$

Le principe du remplissage inverse des eaux donne le débit alloué à la source k [8] :

$$R_k = \frac{1}{2} max \left[\left(\log_2 \frac{\sigma_{X_k}^2}{D} \right), 0 \right] \quad \text{bit} \tag{6.25}$$

où D est la distorsion maximale acceptable. Par suite, l'allocation des bits optimale (dans le cas gaussien et pour l'erreur quadratique) revient à optimiser le paramètre $\lambda = \frac{1}{2} \log_2 D$. Sans perte de généralité, on peut supposer que $\sigma_{X_1}^2 \geq \sigma_{X_2}^2 \geq \cdots \geq \sigma_{X_K}^2$. Si le budget R est alloué aux seuls indices de quantification (par exemple, pour une allocation fixe des bits), on cherche λ et $1 \leq k \leq K$ tels que

$$\sum_{i=1}^{k} \left[\frac{1}{2} \log_2 \left(\sigma_{X_i}^2 \right) - \lambda \right] \leq R. \tag{6.26}$$

L'égalité est obtenue pour

$$\lambda = \frac{\sum_{i=1}^{k} \frac{1}{2} \log_2 \sigma_{X_i}^2 - R}{k}. \tag{6.27}$$

L'optimisation de λ peut donc être réalisée de façon itérative :

- initialiser $k = 1$ et $t = \frac{1}{2}\log_2 \sigma^2_{X_1}$
- tant que $k < K$ et $(t - \frac{k}{2}\log_2 \sigma^2_{X_{k+1}}) < R$
  - $k := k + 1$
  - $t := t + \frac{1}{2}\log_2 \sigma^2_{X_k}$
- calculer $\lambda = (t - R)/k$

Pour la valeur de λ trouvée, l'allocation des bits optimale est alors donnée par l'équation 6.25 avec $\lambda = \frac{1}{2}\log_2 D$.

Ce principe d'allocation des bits est en fait appliqué dans de nombreux systèmes de codage par transformée. Par exemple, dans le standard G.722.1 [39] (codage MLT avec quantification scalaire codée par codes de Huffman vectoriels), l'allocation des bits (appelé catégorisation) est calculée à partir de l'énergie quantifiée de sous-bandes. La quantification étant logarithmique avec un pas de $10\log_{10} 2^{1/2} = 1.50$ dB. L'algorithme d'allocation est conçu pour tenir compte d'une allocation maximale.

Annexe 6.C : Modèle TCX utilisé

Le modèle TCX combine les avantages du modèle CELP et ceux du codage par transformée. Il utilise la prédiction linéaire court–terme et minimise l'erreur de codage dans le domaine perceptuel. Du codage par transformée il exploite la concentration de l'énergie sur un petit nombre de coefficients. Ce modèle contourne la recherche complexe dans le dictionnaire innovateur du CELP [17] et permet de commuter de façon élégante entre un codage CELP et TCX [6] – cette commutation entre CELP et TCX est transparente (*seamless*), contrairement à l'approche de [55].

Le modèle TCX employé ici est montré à la figure 6.23. Dans le contexte du codage ACELP/TCX multi-mode, on applique le modèle sans prédiction de pitch de [12] afin d'améliorer la qualité de l'ACELP principalement pour la musique. Le signal codé correspond ici à une version décimée à 12.8 kHz du signal d'entrée comme dans [36]. On alloue ainsi l'essentiel du budget de bits aux basses fréquences (les plus importantes perceptuellement) et le codage ACELP de l'AMR-WB peut être réutilisé. La longueur des trames du codage TCX est fixée ici à 20, 40 ou 80 ms.

Analyse et quantification LPC : L'analyse LPC est réalisée toutes les 20 ms et est identique à celle de l'AMR-WB [36] sauf que le calcul des autocorrélations utilise un fenêtrage symétrique en cosinus sur 35 ms. Le facteur de pré-accentuation est fixé à 0.68 comme dans l'AMR-WB. Les coefficients de prédiction sont quantifiés sous forme d'ISF avec le quantificateur prédictif MA(1) à 46 bits de l'AMR-WB [36]. Ils sont interpolés dans le domaine des LSF pour chaque sous-trame de 5 ms. Le filtre perceptuel est donné par [36] :

$$\hat{W}(z) = \frac{\hat{A}(z/\gamma)}{1 - \alpha z^{-1}} \tag{6.28}$$

avec $\hat{A}(z)$ est le filtre prédicteur (quantifié) d'ordre 16 et $\alpha = 0.68$.

Fenêtrage et addition-recouvrement : Le modèle TCX de [2] utilise un fenêtrage rectangulaire, si bien que des artefacts peuvent être audibles en bordure de trames en raison d'une discordance de phases et d'amplitudes, et l'analyse fréquentielle est limitée par un fort étalement spectral [20].

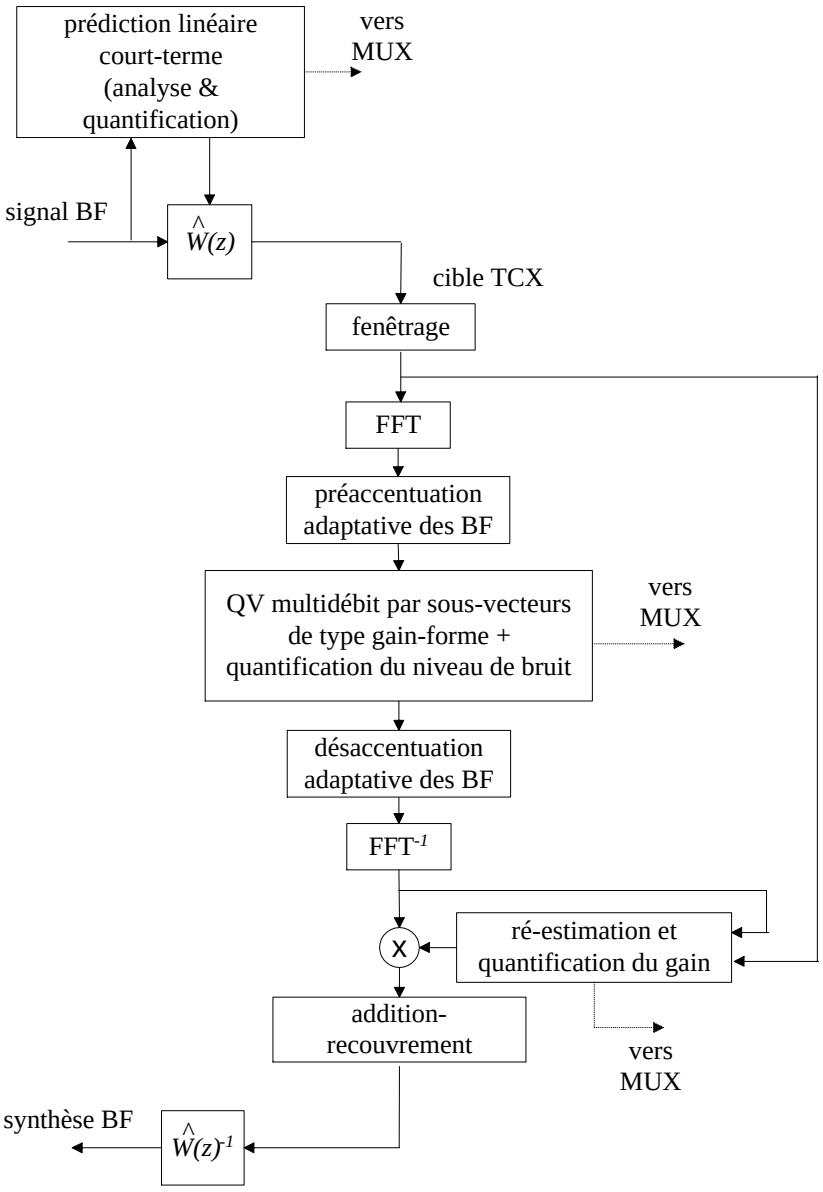


Figure 6.23: Codage TCX sans prédiction de pitch du modèle AMR-WB+.

Pour réduire les effets de bord, un fenêtrage avec recouvrement temporel est utilisé sur la cible avant quantification et la cible est reconstruite par addition-recouvrement. Ce fenêtrage est identique à

celui de [20] ; par contre, il est ici fonction de la longueur de la trame de codage TCX :

$$h(n) = \begin{cases} sin((0.5+n)\pi/2L_1) & \text{si } 0 \leq n \leq L_1 - 1 \\ 1 & \text{si } L_1 \leq n \leq L - L_2 - 1 \\ cos((0.5+n-L+L_2)\pi/2L_2) & \text{si } L - L_2 \leq n \leq L - 1 \end{cases} \tag{6.29}$$

où L est la longueur de la fenêtre, et L_1 et L_2 sont les longueurs de recouvrement à gauche et à droite respectivement. Le recouvrement à gauche dépend de la trame passée. Le recouvrement à droite est de 2.5 ms pour une trame de 20 ms, 5 ms pour une trame 40 ms, et 10 ms pour une trame de 80 ms. La cible étant échantillonnée à 12.8 kHz, la longueur L de la FFT est un multiple de 9. La redondance due au recouvrement est de 11 %.

FFT et FFT inverse : Les coefficients de la FFT $\mathbf{X} = (X_1, \cdots, X_L)$ sont regroupés de sorte que X_1 soit la composante continue, X_2 corresponde à la fréquence normalisée $\pi/2$ et X_{2i}, X_{2i+1} soient respectivement les parties réelles et imaginaires d'un coefficient de Fourier.

Pré-accentuation et désaccentuation adaptative des basses fréquences : Pour corriger certains défauts de codage appaîssant avec des sons spécifiques présentant une courbe de masquage très basse dans les graves, les basses fréquences sont amplifiées de façon adaptative dans l'AMR-WB+. Cet ajustement constitue un écart au principe du codage de la cible TCX suivant l'erreur quadratique.

Quantification vectorielle multi-débit : On applique la technique de codage introduite dans ce chapitre. Pour minimiser la complexité, l'optimisation du gain TCX g est réduite à la seule estimation du gain à partir de l'énergie des sous-vecteurs. Le niveau de bruit σ_{inj} est estimé pendant cette optimisation.

BIBLIOGRAPHIE

[1] T. Moriya and M. Honda, "Transform coding of speech with weighted vector quantization," in *Proc. ICASSP*, 1987, pp. 1629–1632.

[2] J.-P. Adoul and R. Lefebvre, *Wideband Speech Coding*, chapter 8, pp. 289–309, Elsevier (W.B. Kleijn and K.K. Paliwal, eds.), 1995.

[3] M. Xie and J.-P. Adoul, "Embedded algebraic vector quantization (EAVQ) with application to wideband audio coding," in *Proc. ICASSP*, 1996, vol. 1, pp. 240–243.

[4] M.J. Sabin and R.M. Gray, "Product Code Vector Quantizers for Waveform and Voice Coding," vol. 32, pp. 474–488, Jun. 1984.

[5] T.M. Cover and J.A. Thomas, *Elements of Information Theory*, Wiley, 1991.

[6] B. Bessette, R. Salami, C. Laflamme, and R. Lefebvre, "A wideband speech and audio codec at 16/24/32 kbit/s using hybrid ACELP/TCX techniques," in *Proc. IEEE Workshop on Speech Coding*, Jun. 1999, pp. 7–9.

[7] R. Zelinski and P. Noll, "Adaptive Transform Coding of Speech Signals," *IEEE Trans. ASSP*, vol. 25, no. 4, pp. 299–309, Aug. 1977.

[8] N.S. Jayant and P. Noll, *Digital Coding of Waveforms – Principles and Applications to Speech and Video*, Prentice-Hall, 1984.

[9] V. Sanchez and J.-P. Adoul, "Low-delay wideband speech coding using a new frequency domain approach," in *Proc. ICASSP*, April 1993, vol. II, pp. 612–615.

[10] J.-H. Chen, *Low-Delay Coding of Speech*, chapter 6, pp. 209–256, Elsevier (W.B. Kleijn and K.K. Paliwal, eds.), 1995.

[11] R. Lefebvre, R. Salami, C. Laflamme, and J.-P. Adoul, "8 kbit/s coding of speech with 6 ms frame-length," in *Proc. ICASSP*, 1993, vol. 2, pp. 612–615.

[12] R. Lefebvre, R. Salami, C. Laflamme, and J.-P. Adoul, "High quality of wideband audio signals using transform coded excitation (TCX)," in *Proc. ICASSP*, 1994, vol. I, pp. 193–196.

[13] N. Moreau and P. Dymarski, "Successive orthogonalizations in the multistage CELP coder," in *Proc. ICASSP*, 1992, pp. 61–64.

[14] N. Moreau and P. Dymarski, "Selection of Excitation Vectors for the CELP Coders," *IEEE Trans. on Speech and Audio Processing*, pp. 29–41, Jan. 1994.

[15] C. Galand, J. Menez, and M. Rosso, "Adaptive Code Excited Predictive Coding," *IEEE Trans. ASSP*, vol. 5, pp. 858–865, Jun. 1992.

[16] J.-M. LeRoux, R. Lefebvre, and J.-P. Adoul, "Comparison of the wavelet decomposition and the fourier transform in TCX encoding of wideband speech and audio," in *Proc. ICASSP*, 1995, vol. 5, pp. 3083–3086.

[17] R. Salami, R. Lefebvre, and C. Laflamme, "A wideband codec at 16/24 kbit/s with 10 ms frames," in *Proc. IEEE Workshop on Speech Coding for Telecommunications*, Sep. 1997, pp. 103–104.

[18] J.-H. Chen and D. Wang, "Transform Predictive Coding of Wideband Speech Signals," in *Proc. ICASSP*, 1996, pp. 275–278.

[19] J.-H. Chen, "A candidate for the ITU-T's new wideband speech coding standard," in *Proc. ICASSP*, 1997, vol. 2, pp. 1359–1362.

[20] A. Jbira, N. Moreau, and P. Dymarski, "Low delay coding of wideband audio (20 Hz–15 kHz) at 64 kbps," in *Proc. ICASSP*, 1998.

[21] S.A. Ramprashad, "A multimode transform predictive coder (MPTC) for speech and audio," in *Proc. IEEE Workshop on Speech Coding*, 1999, pp. 10–12.

[22] J. Schnitzler, J. Eggers, C. Erdmann, and Peter Vary, "Wideband speech coding using forward/backward adaptive prediction with mixed time/frequency domain excitation," in *Proc. IEEE Workshop on Speech Coding*, 1999, pp. 4–6.

[23] ISO/IEC 14496-3, "Information technology – Coding of audio-visual objects – Part 3: Audio," JTC 1/SC 29, Dec. 2001.

[24] K. Brandenburg, "MP3 and AAC explained," in *Proc. of AES 17th Int. Conf. on High Quality Audio Coding*, Sep. 1999.

[25] M. Bosi et al., "ISO/IEC MPEG-2 Advanced Audio Coding," in *Proc. of the 101st AES Convention*, 1996, Preprint 4382.

[26] M. Dietz and S. Meltzer, "CT-aacPlus – a state-of-the-art Audio Coding Scheme," EBU Technical Review, Jul. 2002.

[27] R. Lefebvre, *Nouveau Modèle de Codage Audio à Prédiction Linéaire et Transformée Orthogonale*, Thèse de doctorat, Université de Sherbrooke, Québec, Canada, Avril 1995.

[28] T. Berger, *Rate Distortion Theory*, Prentice-Hall, 1971.

[29] A. Gersho and R.M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic Publishers, 1992.

[30] R.M. Gray and D.L. Neuhoff, "Quantization," *IEEE Trans. Inform. Theory*, vol. 44, no. 6, pp. 2325–2383, Oct. 1998.

[31] C.K. Tun and D.T. Magill, "The Residual-Excited Linear Prediction Vocoder with Transmission Rate below 9.6 kbit/s," *IEEE Trans. Com.*, vol. COM-23, no. 12, pp. 1466–1473, Dec. 1975.

[32] W.A. Pearlman and R.M. Gray, "Source coding of the Discrete Fourier Transform," *IEEE Trans. Inf. Th.*, vol. 24, pp. 683–692, Nov. 1978.

[33] M. Xie, *Quantification vectorielle algébrique et codage de parole en bande élargie*, Thèse de doctorat, Université de Sherbrooke, Québec, Canada, Février 1996.

[34] J.H. Conway and N.J.A. Sloane, *Sphere Packings, Lattice and Groups*, Springer, 3rd edition, 1999.

[35] K. Sayood, *Introduction to Data Compression, 2nd ed.*, Morgan Kaufman, 2000.

[36] B. Bessette, R. Salami, C. Laflamme, and R. Lefebvre, "The Adaptive Multirate Wideband Speech Codec (AMR-WB)," *IEEE Trans. on Speech and Audio Processing*, vol. 10, no. 8, pp. 620–637, Nov. 2002.

[37] P. Elias, "Universal Codeword Sets and Representations of the Integers," *IEEE Trans. on Inform. Theory*, vol. 21, no. 2, pp. 194–207, March 1975.

[38] S. Ragot, "Draft description of packetization and frame-loss error concealment in amr-wb+," unpublished work, Fev. 2003.

[39] PictureTel Corp., "A wideband audio coder for multimedia communications with a 16 kHz sample rate - a detailed description of the PictureTel candidate algorithm," ITU-T Study Group 16, WP3 Q20/16, May 17–28 1999.

[40] D.E. Knuth, *The Art of Computer Programming, Volume 2: Seminumerical Algorithms (3rd ed.)*, Addison-Wesley, 1997.

[41] Y. Shoham and A. Gersho, "Efficient bit allocation for an arbitrary set of quantizers," *IEEE Trans. ASSP*, vol. 36, no. 9, pp. 1445–1453, Sep. 1988.

[42] E.A. Riskin, "Optimal bit allocation via the generalized BFOS algorithm," *IEEE Trans. Inform. Theory*, vol. 37, pp. 400–402, Mar. 1991.

[43] M.R. Shroeder and B.S. Atal, "Code-excited linear prediction (CELP): High-quality speech at very low bit rates," in *Proc. ICASSP*, 1985, pp. 937–940.

[44] P. Kroon and W.B. Kleijn, *Linear-Prediction based Analysis-by-Synthesis Coding*, chapter 3, pp. 79–119, Elsevier (W.B. Kleijn and K.K. Paliwal, eds.), 1995.

[45] J.-P. Adoul, P. Mabilleau, M. Delprat, and S. Morissette, "Fast CELP coding based on algebraic codes," in *Proc. ICASSP*, 1987, pp. 1957–1960.

[46] G. Davidson, M. Yong, and A. Gersho, "Real-time vector excitation coding of speech at 4800 bps," in *Proc. ICASSP*, 1987, pp. 2189–2192.

[47] ITU-T Recommandation G.729, "Coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited linear-prediction (CS-ACELP," 03/96.

[48] J.-P. Adoul and C. Laflamme, "Dynamic codebook for efficient speech coding based on algebraic codes," International patent WO 91/13432.

[49] J.-P. Adoul and C. Laflamme, "Depth-first algebraic-codebook search for fast coding of speech," International patent WO 96/28810.

[50] J.-P. Adoul and C. Laflamme, "Algebraic codebook with signal-selected pulse amplitudes for fast coding of speech," International patent WO 96/24925.

[51] J.P. Ashley, E.M. Cruz-Zeno, U. Mittal, and W. Peng, "Wideband coding of speech using a scalable pulse codebook," in *Proc. IEEE Workshop on Speech Coding*, Sep. 2000.

[52] R.M. Gallager, *Information Theory and Reliable Communications*, Wiley, 1968.

[53] R.S. Cheng and S. Verdú, "Gaussian multiaccess channels with ISI: capacity region and multi-user waterfilling," *IEEE Trans. Inf. Th.*, vol. 39, no. 3, pp. 773–785, May 1993.

[54] A.N. Kolmogorov, "On the Shannon theory of information transmission in the case of continuous signals," *Trans. IRE*, vol. IT-2, pp. 102–108, 1956.

[55] L. Tancerel, S. Ragot, V.T. Ruoppila, and R. Lefebvre, "Combined speech and audio coding by discrimination," in *Proc. IEEE Workshop on Speech Coding*, Sep. 2000.

[56] D.H. Lehmer, "Mathematical methods in large-scale computing units," in *Proceedings of the Second Symposium on Large-Scale Digital Computing Machinery, Harvard University Press*, 1951, pp. 141–145.

[57] S. Ramprashad, "The multimode transform predictive coding paradigm," *IEEE Trans. on Speech and Audio Processing*, vol. 11, no. 2, pp. 117– 129, Mar. 2003.

[58] P. Loyer, J.-M. Moureaux, and M. Antonini, "Lattice Codebook Enumeration for Generalized Gaussian Source," *IEEE Trans. on Inf. Th.*, vol. 49, no. 2, pp. 521–528, Feb. 2003.

[59] P. Combescure et al., "A 16, 24, 32 kbit/s wideband speech codec based on ATCELP," in *Proc. ICASSP*, 1999, vol. 1, pp. 5–8.

[60] G.D. Forney, "Coset Codes. II. Binary Lattices and related codes," *IEEE Trans. on Inf. Th.*, vol. 34, no. 5, pp. 1152–1187, Sep. 1988.

[61] I.H. Witten, R.M. Neal, and J.G. Cleary, "Arithmetic coding for data compression," *Communications of the ACM*, vol. 30, no. 6, pp. 520–540, Jun. 1987.

Chapitre 7

CONCLUSIONS GÉNÉRALES

Cette thèse s'est donnée pour but d'approfondir et de valider l'intérêt de la quantification vectorielle par réseau de points en codage audio, dans le cadre du développement du modèle ACELP/TCX multi-mode [1, 2]. Le codage de source avec perte est généralement décomposé en deux fonctions : la modélisation et la quantification. Les travaux présentés ici n'ont considéré que la quantification. Les applications visées dans cette thèse sont la quantification des coefficients de prédiction linéaire (ou coefficients LPC) utilisée en codage prédictif de type ACELP, TCX, etc. et la quantification des coefficients de transformée dans le modèle TCX. Dans le premier cas, on sait d'après les travaux de [3, 4, 5, 6, 7, 8] que la quantification par réseau de points permet d'atteindre à débit fixe un compromis performances/complexité aussi (voire plus) intéressant qu'en quantification non-structurée par produit cartésien ou multi-étage. Néanmoins, la quantification non-structurée est généralement adoptée en pratique. La quantification des coefficients LPC n'est en outre qu'un sous-problème du codage audio. A l'opposé, la quantification s'avère être la clé de voûte du modèle TCX ; elle contrôle en effet la qualité de codage, car elle consomme l'essentiel du budget de bits. Dans ce cas, la quantification par réseau de points s'avère un outil original et adapté : les réseaux de points se prêtent idéalement à la quantification multi-débit et à un codage suivant l'erreur quadratique.

Les travaux présentés ici complètent, étendent et remplacent dans une large mesure des tech-

niques présentées dans [4, chaps. 1 et 3] et [9]. Cette thèse a ainsi permis de compléter le codage de Voronoï quasi-ellipsoïdal introduit très partiellement dans [4, chap. 1]. Dans ce cas, les principales contributions ont porté sur un traitement rigoureux du codage de Voronoï quasi-ellipsoïdal, et le développement d'un algorithme d'indexation général et d'un algorithme de recherche de complexité maximale bornée employant une projection et une réduction avec dichotomie. Plusieurs projections sont proposées : sur région de Voronoï à l'aide des vecteurs pertinents ou sur ellipsoïde. Ce codage a été évalué pour une source gaussienne ellipsoïdale et s'est avéré presque aussi efficace (en dimension 8 à un débit de 20 bits par vecteur) que la quantification non-structurée par sous-vecteurs, mais avec un stockage négligeable. Ce codage a en outre été appliqué de façon préliminaire à la quantification de coefficients de prédiction linéaire par analyse en composantes principales et modèles de mélanges de gaussiennes dans [10].

Une contribution significative de cette thèse a porté sur le développement et la description de techniques de quantification vectorielle algebrique multi-débit pour le codage TCX. Une version préliminaire avait été introduite dans [4, chap. 3] et [9] sous une forme dite imbriquée (restreinte à la dimension 8). On a présenté ici plus particulièrement deux techniques : la quantification multi-débit par extension de Voronoï et le codage par troncature de Voronoï. Celles-ci ont l'avantage d'avoir une plus faible complexité que la quantification multi-débit de [4, chap. 3] et [9] (en évitant la coûteuse opération de saturation), tout en étant plus adaptées au codage TCX à débits et longueurs de trames différents. Elles impliquent, comme dans [4, chap. 3] et [9], de coder une information supplémentaire décrivant approximativement l'énergie et spécifiant l'allocation des bits. Ces deux techniques de quantification multi-débit ont été validées avec une source gaussienne sans mémoire ; elles se situent à 2–2.5 dB de la borne débit–distorsion pour un codage dans RE_8, dans l'hypothèse où l'information supplémentaire est codée à un débit égal à son entropie. Pour comparaison, la quantification scalaire entropique se situe à 1.5 dB de cette borne, mais nécessite un codage (implicite ou explicite) du pas et du nombre de niveaux de quantification.

Finalement, le codage algébrique de la cible TCX introduit dans [4, chap. 3] et [9] a été généralisé, pour appliquer la quantification multi-débit décrite précédemment. Le problème du codage TCX est ramené à l'optimisation d'un facteur d'échelle scalaire en vertu du principe d'allocation des

bits par remplissage inverse des eaux. Ce principe d'allocation est optimal sous certaines conditions, en supposant en particulier que les sous-vecteurs ont une distribution gaussienne i.i.d. Néanmoins des tests statistiques ont montré que les sous-vecteurs de la cible TCX ont plutôt une distribution laplacienne. D'un certain point du vue, le codage ACELP/TCX multi-mode peut être vu comme l'équivalent en codage de la quantification par "filet protecteur" (*safety-net vector quantization*) [11]. En effet, le codage ACELP est fortement spécialisé pour les signaux de parole ; le caractère universel du codage ACELP/TCX dépend donc en grande partie du codage TCX, qui joue le rôle de "filet protecteur". La quantification vectorielle par réseau de points a ainsi un avantage important, car elle rend le codage TCX plus flexible et plus général.

Les travaux présentés dans cette thèse peuvent être approfondis et enrichis de plusieurs façons. Dans cette optique, la quantification des coefficients transformés en codage TCX est sans doute la plus prometteuse en terme d'impact technologique. La quantification par réseau de points est encore à ce jour relativement peu utilisée en codage audio, et d'usage anecdotique. Or, la quantification multi-débit décrite ici possède de multiples avantages. En particulier, celle-ci est bien adaptée au codage par transformée multi-débit. Elle pourrait être appliquée à des modèles de codage par transformée de type MPEG-AAC, afin de remplacer le codage scalaire entropique. On pourrait chercher à en développer des variantes ; des comparaisons de performances-complexité devraient être conduites entre différents systèmes de quantification au sein d'un même modèle de codage par transformée (du signal ou de l'excitation). En outre, un codage plus évolué que le codage unaire pourrait être développé pour représenter les numéros de dictionnaires en quantification multi-débit. Sur le plan de la modélisation, on pourrait étudier l'intérêt d'incorporer un modèle perceptuel en codage TCX. En effet, des expériences ont montré qu'en utilisant l'erreur quadratique le codage TCX peut introduire des artefacts en basses fréquences – l'énergie n'étant pas un critère perceptuel viable. On pourrait également explorer l'intérêt du codage hiérarchique ACELP/TCX, puisque qu'avec la quantification multi-débit développée ici le modèle TCX est désormais plus flexible et universel que dans [9]. Enfin, la correction des pertes de trames dans le modèle TCX est un sujet encore ouvert et important pour les applications réelles.

Plus généralement, on pourrait aussi explorer des pistes de recherche, telles que le codage TCX à débit variable (en particulier à qualité constante), le codage audio avec segmentation temporelle adaptative et modèle TCX (ou ACELP/TCX multi-mode) pour des applications de stockage. Les outils développés ici, en particulier la quantification multi-débit, pourraient aussi être appliqués à d'autres signaux tels que l'image ou la vidéo.

BIBLIOGRAPHIE

[1] R. Salami, R. Lefebvre, and C. Laflamme, "A wideband codec at 16/24 kbit/s with 10 ms frames," in *Proc. IEEE Workshop on Speech Coding (Pocono Manor, PA, USA)*, Sep. 1997, pp. 103–104.

[2] B. Bessette, R. Salami, C. Laflamme, and R. Lefebvre, "A wideband speech and audio codec at 16/24/32 kbit/s using hybrid ACELP/TCX techniques," in *Proc. IEEE Workshop on Speech Coding*, June 1999, pp. 7–9.

[3] M. Xie and J.-P. Adoul, "Fast and low-complexity LSF quantization using algebraic vector quantizer," in *Proc. ICASSP*, 1995, vol. 1, pp. 716–720.

[4] M. Xie, *Quantification vectorielle algébrique et codage de parole en bande élargie*, Thèse de doctorat, Université de Sherbrooke, Québec, Canada, Fév. 1996.

[5] J. Pan and T.R. Fischer, "Vector quantization-lattice vector quantization of speech LPC coefficients," in *Proc. ICASSP*, 1994, vol. 1, pp. 513–516.

[6] J. Pan, "Two-stage vector quantization-pyramidal lattice vector quantization and application to speech lsp coding," in *Proc. ICASSP*, 1996, vol. 2, pp. 737–740.

[7] S. Ragot, J.-P. Adoul, R. Lefebvre, and R. Salami, "Low complexity LSF quantization for wideband speech coding," in *Proc. IEEE Workshop on Speech Coding*, 1999, pp. 22–24.

[8] S. Ragot, R. Lefebvre, R. Salami, and J.-P. Adoul, "Stochastic-algebraic wideband LSF quantization," in *Proc. ICASSP*, 2000, vol. II, pp. 1169–1172.

[9] M. Xie and J.-P. Adoul, "Embedded algebraic vector quantizers (EAVQ) with application to wideband speech coding," in *Proc. ICASSP*, 1996, vol. 1, pp. 240–243.

[10] S. Ragot, H. Lahdili, and R. Lefebvre, "Wideband LSF quantization by generalized Voronoi codes," in *Proceedings of Eurospeech, Aalborg, Denmark*, Sep. 2001, pp. 2319–2322.

[11] T. Eriksson, J. Lindén, and J. Skoglund, "A safety-net approach for improved exploitation of speech correlations," in *Proc. Int. Conf. on Digital Signal Processing*, 1995, vol. 1, pp. 96–101.