ISSN xxxx-xxxx

Outils pour la compression
Application à la compression des signaux audio

N. MOREAU

N. MOREAU

Novembre 2009

Table des matières

Introduction

Dans notre vie courante, nous sommes amenés à manipuler fréquemment des signaux sous forme comprimée : il suffit que l'on utilise un téléphone mobile, un baladeur "mp3", un appareil photographique numérique ou un lecteur DVD. En effet les signaux impliqués dans ces applications, respectivement du signal de parole en bande téléphonique, de la musique en bande Hi-Fi, des images fixes ou du signal vidéo, ont non seulement subi une opération d'échantillonnage et de quantification pour leur donner une forme apte à les stocker dans des mémoires de masse ou à les transférer dans des réseaux mais ils ont aussi subi une opération de compression. La première opération est très basique et elle est présentée dans tous les cours ou les livres de base en Traitement du Signal. La seconde opération est plus spécifique et fait l'objet de ce livre : on présentera les outils standards réalisant cette compression puis on montrera comment ces outils sont exploités pour comprimer des signaux de parole ou de musique. Dans une première partie, on focalisera l'attention sur un problème à caractère plutôt théorique, il s'agira de minimiser une erreur quadratique moyenne, la seconde partie sera plus concrète, elle nuancera la démarche précédente en cherchant à minimiser un débit d'information en respectant une contrainte issue de la psychoacoustique. On verra qu'effectivement comprimer du signal consiste non seulement à chercher à supprimer toute redondance dans le signal d'origine mais aussi à tenter d'éliminer toute information non perçue.

Les techniques de compression présentées dans ce livre ne sont pas nouvelles. Elles ont été élaborées dans un cadre théorique, la théorie de l'information et le codage de source, visant à formaliser le premier (et le dernier) élément d'une chaîne de communication numérique : le passage d'un signal analogique (à temps continu et à valeurs continues) vers un signal numérique (à temps discret et à valeurs discrètes). Elles sont issues des travaux de C. Shannon publiés au début des années 50. Par contre, excepté le développement de codeurs de parole dans les années 70 pour promouvoir un réseau téléphonique commuté entièrement numérique, ces techniques sont devenues vraiment utilisées uniquement vers la fin des années 80 sous l'influence de groupes de travail, par exemple le "Groupe Spécial Mobile (GSM)", le "Joint Photographic Experts Group (JPEG)" et le "Moving Picture Experts Group (MPEG)".

Les résultats obtenus sont assez spectaculaires et ont permis les applications citées précédemment. Qu'on en juge en prenant l'exemple d'un signal de musique. On sait qu'il peut être reconstruit avec une qualité quasi parfaite (qualité "CD") s'il a été échantillonné à une fréquence de 44.1 kHz puis quantifié avec une résolution de 16 bits. Transferré dans un réseau, le débit nécessaire pour une voie mono est alors de 705 kbit/s. Le codeur audio le plus abouti, le codeur MPEG-4 AAC, assure la "transparence" à un débit de l'ordre de 64 kbit/s soit un taux de compression supérieur à 10 et le tout nouveau codeur MPEG-4 HE-AACv2 normalisé en 2004 assure une qualité très convenable (pour de la télévision sur mobiles) à 24 kbit/s pour 2 voies stéréo. Le taux de compression devient supérieur à 50 !

Dans ce livre, on présentera d'abord les outils standards (quantificateur scalaire, quantificateur prédictif, quantificateur vectoriel, techniques de codage par transformée, en sous-bandes, codage entropique). Pour pouvoir comparer les performances de ces outils, on prendra un exemple académique, celui de la quantification d'une réalisation $x(n)$ d'un processus aléatoire mono-

dimensionnel $X(n)$. C'est une approche théorique mais elle permettra non seulement de donner des mesures objectives de ces performances mais aussi de mettre en évidence la cohérence qui existe entre tous les outils disponibles. Dans une deuxième partie, on s'intéressera à la compression des signaux audio (parole en bande téléphonique, en bande élargie, musique en bande Hi-Fi).

Dans tout ce livre, on fera appel à des notions de base en Traitement du Signal en utilisant le vocabulaire et les notations suivantes. On considérera un processus aléatoire mono-dimensionnel $X(n)$ stationnaire, centré, de puissance σ_X^2 et de densité spectrale de puissance $S_X(f)$. On le supposera également gaussien d'abord parce que le caractère gaussien est maintenu par toute transformation linéaire en particulier un filtrage ce qui simplifie beaucoup le formalisme et aussi parce qu'on verra qu'un signal gaussien est le signal le plus difficile à coder, celui qui entraîne la puissance de l'erreur de quantification la plus élevée. On notera $\underline{X}(m)$ le vecteur colonne de dimension N construit à partir de $X(mN)\cdots X(mN+N-1)$. Ces N variables aléatoires sont statistiquement complètement définies par leur densité de probabilité conjointe

$$p_X(\underline{x}) = \frac{1}{(2\pi)^{N/2}\sqrt{\det \mathbb{R}_X}} \exp(-\frac{1}{2}\underline{x}^t \mathbb{R}_X^{-1} \underline{x})$$

où $\mathbb{R}_X$ est la matrice d'autocovariance

$$\mathbb{R}_X = \mathbb{E}\{\underline{X}(m)\underline{X}^t(m)\} = \begin{bmatrix} r_X(0) & r_X(1) & \cdots & r_X(N-1) \\ r_X(1) & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & r_X(1) \\ r_X(N-1) & \cdots & r_X(1) & r_X(0) \end{bmatrix}$$

matrice de Toeplitz de dimension $N \times N$. En outre, on supposera le processus $X(n)$ autorégressif d'ordre P, c'est à dire obtenu par filtrage d'un bruit blanc centré $W(n)$ de variance σ_W^2 par un filtre d'ordre P de fonction de transfert $1/A(z)$ avec $A(z)$ de la forme

$$A(z) = 1 + a_1 z^{-1} + \cdots + a_P z^{-P}.$$

L'intérêt de prendre comme exemple la quantification d'un processus autorégressif, est de pouvoir exprimer simplement toutes les caractéristiques statistiques de la source en fonction des paramètres du filtre comme, par exemple, sa densité spectrale de puissance

$$S_X(f) = \frac{\sigma_W^2}{|A(f)|^2}$$

où la notation $A(f)$ procède d'un abus d'écriture puisque, plus proprement, on devrait écrire $A(\exp(j2\pi f))$. L'intérêt est aussi de pouvoir donner l'expression analytique de la puissance de l'erreur de quantification pour différents schémas de quantification lorsque l'on choisit comme mesure de distorsion l'erreur quadratique. Une comparaison des performances des différents schémas de quantification est alors possible. D'un point de vue pratique, cet exemple n'est pas absurde puisqu'il est un modèle raisonnable pour un certain nombre de signaux, par exemple pour du signal de parole (la stationnarité n'est alors vraie que de façon locale) lorsque l'on choisit un ordre P suffisamment élevé (8 ou 10 par exemple).

Première partie

Outils pour la compression des signaux

Chapitre 1

Quantification scalaire

1.1 Introduction

Considérons un signal à temps discret $x(n)$ prenant ses valeurs dans l'intervalle $[-A, +A]$. Définir un quantificateur scalaire avec une résolution de b bits par échantillon consiste à réaliser trois opérations :

1. une partition de l'intervalle $[-A, +A]$ en $L = 2^b$ intervalles distincts $\{\Theta^1 \cdots \Theta^L\}$ de longueurs $\{\Delta^1 \cdots \Delta^L\}$,
2. une numérotation des éléments de la partition $\{i^1 \cdots i^L\}$,
3. la sélection d'un représentant par intervalle, l'ensemble de ces représentants composant un dictionnaire (codebook)[1] $C = \{\hat{x}^1 \cdots \hat{x}^L\}$.

La procédure d'encodage (à l'émetteur) consiste à décider à quel élément de la partition appartient $x(n)$ puis à lui associer le numéro $i(n) \in \{1 \cdots L = 2^b\}$ correspondant. C'est le numéro de l'intervalle choisi, le symbole canal, qui sera transmis ou stocké. La procédure de décodage (au récepteur) consiste à associer au numéro $i(n)$ le représentant correspondant $\hat{x}(n) = \hat{x}^{i(n)}$ choisi parmi l'ensemble des représentants $\{\hat{x}^1 \cdots \hat{x}^L\}$. Formellement, on peut observer qu'un quantificateur est une application non bijective de $[-A, +A]$ dans un ensemble fini C plus une règle d'affectation

$$\hat{x}(n) = \hat{x}^{i(n)} \in \{\hat{x}^1 \cdots \hat{x}^L\} \quad \text{ssi} \quad x(n) \in \Theta^i.$$

C'est un processus irréversible entraînant une perte d'information, une erreur de quantification que l'on notera $q(n) = x(n) - \hat{x}(n)$. Il est nécessaire de définir une mesure de distorsion $d[x(n), \hat{x}(n)]$. On choisira par la suite la mesure de distorsion la plus simple, l'erreur quadratique

$$d[x(n), \hat{x}(n)] = |x(n) - \hat{x}(n)|^2.$$

C'est une mesure de distorsion ponctuelle (par lettre). Il est nécessaire de définir une mesure de distorsion plus globale. On prendra l'erreur quadratique moyenne (EQM)

$$D = \mathbb{E}\{|X(n) - \hat{x}(n)|^2\}.$$

Cette erreur est donc simplement la puissance de l'erreur de quantification. On la notera par la suite σ_Q^2.

Les tracés de la figure 1.1 montrent à gauche le signal avant quantification et les éléments d'une partition de l'intervalle $[-A, +A]$ lorsque $b = 3$ et à droite l'ensemble des représentants correspondants et le signal reconstruit. On observe aussi le comportement de l'erreur de quanti-

[1] Dans le cas scalaire, on parle habituellement de niveaux de quantification, de pas de quantification, de seuil de décision. On adoptera directement le vocabulaire propre à la quantification vectorielle.

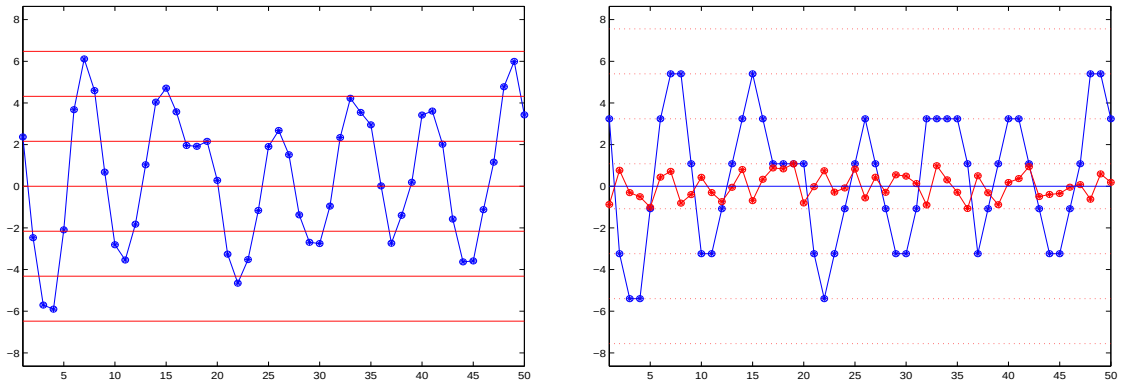

FIG. 1.1 – Signal avant quantification et partition de l'intervalle $[-A, +A]$ à gauche et ensemble des représentants, signal reconstruit et erreur de quantification à droite.

fication. N'est pas matérialisé ici le "flux binaire" circulant entre l'émetteur et le récepteur.

Le problème consiste à définir maintenant le quantificateur optimal c'est à dire à déterminer les éléments de la partition $\{\Theta^1 \cdots \Theta^L\}$ et l'ensemble des représentants $\{\hat{x}^1 \cdots \hat{x}^L\}$ minimisant σ_Q^2.

1.2 Quantification scalaire optimale

On suppose que $x(n)$ est la réalisation d'un p.a. $X(n)$ stationnaire à valeurs réelles. Pour la quantification scalaire, ce qui compte c'est uniquement la distribution des valeurs que prend le p.a. $X(n)$ à l'instant n. Aucune exploitation directe de la corrélation existant entre les valeurs que prend le processus à des instants différents n'est possible. La connaissance de la densité de probabilité marginale de $X(n)$ est donc suffisante. On la notera $p_X(.)$.

1.2.1 Conditions nécessaires d'optimalité

Pour caractériser le quantificateur scalaire optimal, il faut trouver la partition et les représentants minimisant

$$\sigma_Q^2 = \mathbb{E}\{[X(n) - \hat{x}(n)]^2\} = \sum_{i=1}^{L} \int_{u \in \Theta^i} (u - \hat{x}^i)^2 p_X(u) du. \tag{1.1}$$

Cette minimisation conjointe n'admet pas de solution simple. Par contre deux conditions nécessaires d'optimalité sont faciles à obtenir. Si on connaît les représentants $\{\hat{x}^1 \cdots \hat{x}^L\}$, on peut calculer la meilleure partition $\{\Theta^1 \cdots \Theta^L\}$. Si on se donne la partition, on peut en déduire les meilleurs représentants. On peut dire que la partie encodage du quantificateur doit être optimale étant donnée la partie décodage et réciproquement. Ces deux conditions nécessaires d'optimalité s'obtiennent simplement lorsque l'on choisit, comme mesure de distorsion, l'erreur quadratique.

1. Etant donné un dictionnaire $\{\hat{x}^1 \cdots \hat{x}^L\}$, la meilleure partition est celle qui vérifie

$$\Theta^i = \{x : (x - \hat{x}^i)^2 \leq (x - \hat{x}^j)^2 \quad \forall j \in \{1 \cdots L\}\ \}.$$

C'est la règle dite du plus proche voisin.

En effet, si l'on appelle t^i la valeur définissant la frontière entre les partitions Θ^i et Θ^{i+1}, la minimisation de l'erreur quadratique moyenne σ_Q^2 relativement à t^i est obtenue en écrivant

$$\frac{\partial}{\partial t^i}[\int_{t^{i-1}}^{t^i} (u - \hat{x}^i)^2 p_X(u) du + \int_{t^i}^{t^{i+1}} (u - \hat{x}^{i+1})^2 p_X(u) du] = 0$$

$$(t^i - \hat{x}^i)^2 p_X(t^i) - (t^i - \hat{x}^{i+1})^2 p_X(t^i) = 0$$

soit

$$t^i = \frac{\hat{x}^i + \hat{x}^{i+1}}{2}.$$

2. Etant donnée une partition $\{\Theta^1 \cdots \Theta^L\}$, les meilleurs représentants sont obtenus par la condition dite du centroïde (ou centre de gravité) de la partie de la densité de probabilité placée dans la région Θ^i

$$\hat{x}^i = \frac{\int_{u\in\Theta^i} u p_X(u)du}{\int_{u\in\Theta^i} p_X(u)du} = \mathbb{E}\{X|X \in \Theta^i\}. \tag{1.2}$$

En effet, on remarque d'abord que la minimisation de σ_Q^2 relativement à $\hat{x}^i$ ne fait intervenir qu'un élément de la somme donnée par (1.1). En écrivant ensuite

$$\frac{\partial}{\partial \hat{x}^i} \int_{u\in\Theta^i} (u - \hat{x}^i)^2 p_X(u)du = 0$$

$$-2\int_{u\in\Theta^i} u p_X(u)du + 2\hat{x}^i \int_{u\in\Theta^i} p_X(u)du = 0$$

on obtient la première égalité de la formule (1.2).

Comme

$$\int_{u\in\Theta^i} u p_X(u)du = \int_{u\in\Theta^i} p_X(u)du \int_{-\infty}^{\infty} u p_{X|\Theta^i}(u)du$$

où $p_{X|\Theta^i}$ est la densité de probabilité conditionnelle de X sachant que $X \in \Theta^i$, on obtient

$$\hat{x}^i = \int_{-\infty}^{\infty} u p_{X|\Theta^i}(u)du$$

$$\hat{x}^i = \mathbb{E}\{X|X \in \Theta^i\}.$$

La valeur que l'on doit choisir est la valeur moyenne de X dans l'intervalle considéré[2].

On peut montrer que ces deux conditions d'optimalité ne sont pas suffisantes pour garantir l'optimalité du quantificateur sauf dans le cas d'une distribution gaussienne.

On remarque que la connaissance explicite de la partition n'est pas nécessaire. Cette partition est entièrement déterminée par la connaissance de la mesure de distorsion, par l'application de la règle du plus proche voisin et par l'ensemble des représentants. Le schéma de l'encodeur et du décodeur est donné figure 1.2.

1.2.2 Puissance de l'erreur de quantification

Lorsque le nombre L de niveaux de quantification est élevé, il est possible d'obtenir explicitement, contrairement au cas précédent, l'expression de la partition optimale et de la puissance de l'erreur de quantification uniquement en fonction de la densité de probabilité $p_X(x)$. Cette hypothèse dite de *haute résolution* veut dire que la densité de probabilité peut être supposée constante dans l'intervalle $[t^{i-1}, t^i]$ et que le représentant peut être pris au milieu de l'intervalle. On peut donc écrire

$$p_X(x) \approx p_X(\hat{x}^i) \quad \text{pour} \quad x \in [t^{i-1}, t^i[$$

$$\hat{x}^i \approx \frac{t^{i-1} + t^i}{2}.$$

[2]Ce résultat a une interprétation en mécanique : le moment d'inertie d'un objet par rapport à un point est minimum lorsque ce point est le centre de gravité.

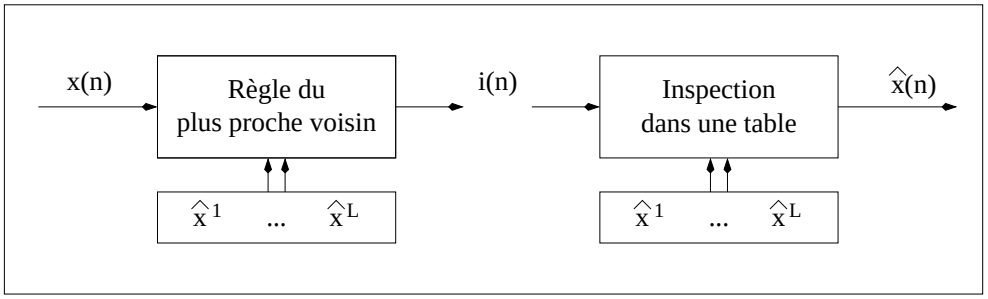

FIG. 1.2 – Encodeur et décodeur.

On appelle

$$\Delta(i) = t^i - t^{i-1}$$

la longueur de l'intervalle $[t^{i-1}, t^i]$ et

$$P_{rob}(i) = P_{rob}\{X \in [t^{i-1}, t^i]\} = p_X(\hat{x}^i)\Delta(i)$$

la probabilité que $X(n)$ appartienne à l'intervalle $[t^{i-1}, t^i]$.

La puissance de l'erreur de quantification s'écrit

$$\sigma_Q^2 = \sum_{i=1}^{L} p_X(\hat{x}^i) \int_{t^{i-1}}^{t^i} (u - \hat{x}^i)^2 du.$$

Comme

$$\int_{t^{i-1}}^{t^i} (u - \hat{x}^i)^2 du = \int_{-\Delta(i)/2}^{+\Delta(i)/2} u^2 du = \frac{\Delta^3(i)}{12}$$

on obtient

$$\sigma_Q^2 = \frac{1}{12} \sum_{i=1}^{L} p_X(\hat{x}^i)\Delta^3(i). \tag{1.3}$$

Elle s'écrit aussi

$$\sigma_Q^2 = \sum_{i=1}^{L} P_{rob}(i)\frac{\Delta^2(i)}{12} = \mathbb{E}\{\frac{\Delta^2}{12}\}.$$

La puissance de l'erreur de quantification ne dépend que de la longueur des intervalles $\Delta(i)$. On cherche $\{\Delta(1) \cdots \Delta(L)\}$ minimisant σ_Q^2. Posons

$$\alpha^3(i) = p_X(\hat{x}^i)\Delta^3(i).$$

Comme

$$\sum_{i=1}^{L} \alpha(i) = \sum_{i=1}^{L} [p_X(\hat{x}^i)]^{1/3}\Delta(i) \approx \int_{-\infty}^{+\infty} [p_X(u)]^{1/3} du = cste$$

puisque cette intégrale ne dépend plus des $\Delta(i)$, on cherche à minimiser la somme des cubes de L nombres positifs ayant une somme constante. Il suffit de les prendre tous égaux. On a donc

$$\alpha(1) = \cdots = \alpha(L)$$

ce qui implique

$$\alpha^3(1) = \cdots = \alpha^3(L)$$

$$p_X(\hat{x}^1)\Delta^3(1) = \cdots = p_X(\hat{x}^L)\Delta^3(L).$$

Cette relation veut dire qu'un intervalle sera d'autant plus petit que la probabilité que $X(n)$ appartienne à cet intervalle sera élevée et que tous les intervalles auront la même contribution dans la puissance de l'erreur de quantification. La puissance de l'erreur de quantification a pour expression

$$\sigma_Q^2 = \frac{L}{12}\alpha^3$$

avec

$$\alpha = \frac{1}{L}\int_{-\infty}^{+\infty}[p_X(u)]^{1/3}du.$$

On obtient donc

$$\sigma_Q^2 = \frac{1}{12L^2}\left(\int_{-\infty}^{+\infty}[p_X(u)]^{1/3}du\right)^3.$$

Comme $L = 2^b$, on obtient la formule dite de Benett

$$\sigma_Q^2 = \frac{1}{12}\left(\int_{-\infty}^{+\infty}[p_X(u)]^{1/3}du\right)^3 2^{-2b}. \tag{1.4}$$

Cette démonstration n'est pas rigoureuse mathématiquement. Elle sera reprise à la fin du chapitre 4 lorsque l'on comparera ce mode de quantification à ce que l'on appellera le quantificateur avec contrainte entropique.

Deux cas particuliers sont intéressants. Lorsque $X(n)$ est distribué suivant une loi uniforme, on obtient

$$\sigma_Q^2 = \frac{A^2}{3}\,2^{-2b} = \sigma_X^2\,2^{-2b}.$$

On pourrait remarquer que le passage par la formule de Benett n'est pas indispensable. On peut obtenir ce résultat directement !

Pour une source gaussienne, centrée, de puissance σ_X^2, pour laquelle

$$p_X(x) = \frac{1}{\sqrt{2\pi\sigma_X^2}}\exp(-\frac{x^2}{2\sigma_X^2})$$

on a

$$\int_{-\infty}^{+\infty}[p_X(u)]^{1/3}du = \int_{-\infty}^{+\infty}\frac{1}{(2\pi\sigma_X^2)^{1/6}}\exp(-\frac{x^2}{6\sigma_X^2})du$$

$$\int_{-\infty}^{+\infty}[p_X(u)]^{1/3}du = (2\pi\sigma_X^2)^{1/3}\sqrt{3}\int_{-\infty}^{+\infty}\frac{1}{(2\pi 3\sigma_X^2)^{1/2}}\exp(-\frac{x^2}{6\sigma_X^2})du$$

$$\int_{-\infty}^{+\infty}[p_X(u)]^{1/3}du = (2\pi\sigma_X^2)^{1/3}\sqrt{3}.$$

On en déduit que

$$\sigma_Q^2 = \frac{1}{12}\,2\pi\sigma_X^2\,3^{3/2}\,2^{-2b}$$

$$\boxed{\sigma_Q^2 = c\;\sigma_X^2\;2^{-2b}} \tag{1.5}$$

avec

$$c = \frac{\sqrt{3}}{2}\pi.$$

Cette formule va servir de référence dans toute la suite de cet ouvrage. Elle admet l'écriture équivalente suivante

$$10 \log_{10} \frac{\sigma_X^2}{\sigma_Q^2} = 6.05\ b - 4.35 \text{ dB}.$$

On en déduit la règle des "6 dB par bit". On peut montrer que pour toute autre distribution (laplacienne, etc.), la puissance minimale de l'erreur de quantification est toujours comprise entre ces deux valeurs. Le cas de la loi uniforme est le cas le plus favorable, le cas de la loi gaussienne est le cas le plus défavorable. Les travaux de Shannon et la théorie "débit/distorsion" formalisent cette constatation.

Il serait intéressant de connaître les propriétés statistiques de l'erreur de quantification. On peut montrer que l'erreur de quantification est décorrélée avec le signal reconstruit mais que cette propriété n'est pas vraie avec le signal original. On peut montrer également que, uniquement dans le cadre de l'hypothèse de haute-résolution, l'erreur de quantification est modélisable par un bruit blanc. Une analyse détaillée est possible (se reporter à l'excellent papier [1]).

1.2.3 Compléments

Algorithme de Lloyd-Max

Dans la pratique, on ne connaît pas $p_X(x)$. Pour construire un quantificateur, on utilise des données empiriques, une base d'apprentissage, en associant à chaque valeur le même poids et on applique l'algorithme de Lloyd-Max, généralement dans sa variante dite LBG. Cet algorithme ayant été généralisé au cas vectoriel, il sera présenté au chapitre suivant.

Transformation non-linéaire

Un quantificateur scalaire non-uniforme peut-être vu comme un quantificateur scalaire uniforme précédé d'une transformation non-linéaire et suivi de la transformation inverse[3]. On définit la transformation par sa caractéristique $f(x)$. Dans cette optique, le problème consiste à choisir la transformation non-linéaire qui minimise la puissance de l'erreur de quantification. Cette étude fait l'objet d'un développement conséquent dans les deux ouvrages de N. Jayant et P. Noll [2] et d'A. Gersho et R. Gray [3]. Ce développement ne me paraît plus fondamental depuis que le quantificateur vectoriel est devenu le véritable outil de base.

Facteur d'échelle

Lors de la mise en œuvre d'une procédure de quantification sur des signaux réels (parole, musique, image), un problème important est l'estimation du paramètre A qui varie avec le temps, les signaux réels ne vérifiant pas l'hypothèse de stationnarité ! On examinera ce problème au chapitre suivant en introduisant une quantification particulière dite "gain/shape" particulièrement bien adaptée aux signaux qui ont des variations de puissance instantanée importantes par exemple les signaux audio.

1.3 Quantification scalaire prédictive

1.3.1 Principe

Dans le développement précédent, on a vu que, pendant l'opération de quantification, aucune exploitation des liens statistiques pouvant exister entre les valeurs successives du signal n'est réalisée. Nous allons voir que le quantificateur scalaire prédictif cherche à décorréler le signal

[3] On utilise habituellement le néologisme "companding" pour compressing + expanding.

avant de le quantifier et que l'exploitation de la corrélation améliore le comportement global du système, c'est à dire diminue la puissance de l'erreur due à la quantification.

Le schéma de principe du quantificateur scalaire prédictif est donné figure 1.3. On retranche

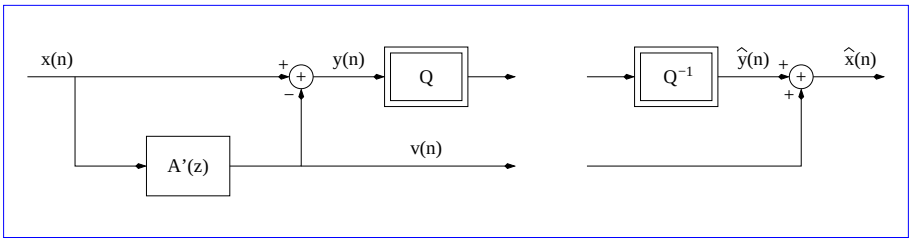

FIG. 1.3 – Schéma de principe du quantificateur prédictif.

au signal $x(n)$ un nouveau signal $v(n)$. On réalise ensuite une procédure d'encodage/décodage sur le signal $y(n) = x(n) - v(n)$. Au décodeur, on rajoute $v(n)$ à la valeur reconstruite $\hat{y}(n)$. On peut tout de suite remarquer que, dans une application réelle de codage, ce schéma n'est pas très réaliste puisque il faudrait alors transmettre aussi au décodeur le signal $v(n)$ mais attendons la fin de ce chapitre pour montrer comment on passe d'un schéma dit *en boucle ouverte* à un schéma *en boucle fermée* plus réaliste mais plus compliqué à analyser.

Si on retranche une valeur au signal avant encodage pour la rajouter après décodage, l'erreur de quantification $q(n) = y(n) - \hat{y}(n)$ et l'erreur de reconstruction $\bar{q}(n) = x(n) - \hat{x}(n)$ sont nécessairement égales à chaque instant puisque

$$q(n) = y(n) - \hat{y}(n) = x(n) - v(n) - [\hat{x}(n) - v(n)] = \bar{q}(n).$$

Leurs puissances respectives sont donc identiques. Comme ce qui intéresse l'utilisateur du système complet est d'avoir une erreur de reconstruction de puissance la plus faible possible, le problème se ramène simplement à chercher à minimiser la puissance de l'erreur de quantification. Si on suppose réalisée une quantification scalaire optimale de $y(n)$, on sait que la puissance de l'erreur de quantification a pour expression

$$\sigma_Q^2 = c\ \sigma_Y^2\ 2^{-2b}.$$

On en conclut que chercher à minimiser la puissance de l'erreur de reconstruction $\sigma_{\bar{Q}}^2$ se ramène à chercher à minimiser la puissance σ_Y^2 de $y(n)$.

On a une grande liberté dans le choix de $v(n)$. Si on prend $v(n)$ sous la forme

$$v(n) = -\sum_{i=1}^{P} a_i x(n-i)$$

en introduisant P paramètres, on parle de *prédiction linéaire à l'ordre P*. Le signal $y(n)$ est l'erreur de prédiction. Il a pour expression

$$y(n) = x(n) - v(n) = x(n) + \sum_{i=1}^{P} a_i x(n-i).$$

La relation entre $x(n)$ et $y(n)$ correspond à une opération de filtrage de fonction de transfert[4]

$$B(z) = 1 + a_1 z^{-1} + \cdots + a_P z^{-P}.$$

[4]Malgré des notations identiques pour éviter de les alourdir, on ne confondra pas les coefficients a_i et l'ordre P du filtre générateur de $x(n)$ et les coefficients et l'ordre du polynôme prédicteur. Dans tout ce chapitre, il ne sera question que du filtre prédicteur.

La minimisation de σ_Y^2 porte sur les coefficients de ce filtre prédicteur.

Ce problème a fait l'objet de nombreuses études à partir des années 60. Tous les livres actuels présentant les techniques de base en traitement du signal incluent un chapitre traitant de ce problème. On pourra, par exemple, consulter le livre de S. Kay [4]. On ne fera ici que quelques rappels succincts.

1.3.2 Quelques rappels sur la théorie de la prédiction linéaire

Introduction : minimisation au sens des moindres carrés

Comme cette théorie, utilisable actuellement dans de nombreuses applications en traitement du signal, a justement été développée dans le bût de déterminer des codeurs de parole à débit réduit et que, dans un codeur de parole, le traitement est alors par bloc de N échantillons[5], le problème peut se poser de la façon suivante : connaissant $\underline{x} = [x(0) \cdots x(N-1)]^t$ déterminer $\underline{a} = [a_1 \cdots a_P]^t$ minimisant la puissance (empirique) de l'erreur de prédiction

$$\underline{a}^{opt} = \arg\min_{\underline{a}} \hat{\sigma}_Y^2$$

avec

$$\hat{\sigma}_Y^2 = \frac{1}{N}\sum_{n=0}^{N-1} y^2(n) = \frac{1}{N}\underline{y}^t\ \underline{y}.$$

Comme

$$\begin{bmatrix} y(0) \\ y(1) \\ \vdots \\ y(N-1) \end{bmatrix} = \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix} + \begin{bmatrix} x(-1) & \cdots & x(-P) \\ x(0) & \cdots & x(-P+1) \\ \vdots & \ddots & \vdots \\ x(N-2) & \cdots & x(N-P-1) \end{bmatrix} \begin{bmatrix} a_1 \\ \vdots \\ a_P \end{bmatrix}$$

soit

$$\underline{y} = \underline{x} + \Gamma\underline{a}$$

on peut écrire

$$\hat{\sigma}_Y^2 = \frac{1}{N}(\underline{x} + \Gamma\underline{a})^t(\underline{x} + \Gamma\underline{a}) = \frac{1}{N}(\underline{x}^t\underline{x} + 2(\Gamma^t\underline{x})^t\underline{a} + \underline{a}^t\Gamma^t\Gamma\underline{a}).$$

Le vecteur $\underline{a}^{opt}$ est celui qui vérifie

$$\frac{\partial}{\partial\underline{a}}\hat{\sigma}_Y^2 = 2\Gamma^t\underline{x} + 2\Gamma^t\Gamma\underline{a}^{opt} = \underline{0}.$$

Si $\Gamma^t\Gamma$ est inversible, on obtient

$$\underline{a}^{opt} = -(\Gamma^t\Gamma)^{-1}\Gamma^t\underline{x}.$$

Le minimum a pour expression

$$\hat{\sigma}_Y^2 = \frac{1}{N}[\underline{x}^t\underline{x} + 2(\Gamma^t\underline{x})^t\underline{a}^{opt} + (\underline{a}^{opt})^t(-\Gamma^t\underline{x})] = \frac{1}{N}\underline{x}^t\underline{x} + \frac{1}{N}(\Gamma^t\underline{x})^t\underline{a}^{opt}.$$

[5] comme on le verra par la suite dans la deuxième partie de ce livre

Approche théorique

On suppose que le signal $x(n)$ peut être interprété (modélisé) comme la réalisation d'un p.a. stationnaire de fonction d'autocovariance $r_X(k) = E\{X(n)X(n-k)\}$. On recherche le vecteur $\underline{a}$ minimisant la puissance de l'erreur de prédiction

$$\sigma_Y^2 = \mathbb{E}\{Y^2(n)\} = \mathbb{E}\{[X(n) + \sum_{i=1}^{P} a_i X(n-i)]^2\}$$

$$\sigma_Y^2 = \mathbb{E}\{X^2(n)\} + 2\sum_{i=1}^{P} a_i \mathbb{E}\{X(n)X(n-i)\} + \sum_{i=1}^{P}\sum_{j=1}^{P} a_i a_j \mathbb{E}\{X(n-i)X(n-j)\}$$

$$\sigma_Y^2 = \sigma_X^2 + 2\underline{a} \begin{bmatrix} r_X(1) \\ \vdots \\ r_X(P) \end{bmatrix} + \underline{a}^t \begin{bmatrix} r_X(0) & \cdots & r_X(P-1) \\ \vdots & \ddots & \vdots \\ r_X(P-1) & \cdots & r_X(0) \end{bmatrix} \underline{a}$$

$$\sigma_Y^2 = \sigma_X^2 + 2\underline{r}^t\underline{a} + \underline{a}^t\mathbb{R}\underline{a}.$$

La minimisation de σ_Y^2 relativement à $\underline{a}$ entraîne les "équations normales"

$$\mathbb{R}\underline{a}^{opt} = -\underline{r}$$

et comme la matrice d'autocovariance $\mathbb{R}$ est définie positive (excepté le cas limite où $X(n)$ est un p.a. harmonique), elle est inversible. On a donc

$$\underline{a}^{opt} = -\mathbb{R}^{-1}\underline{r}. \tag{1.6}$$

On a aussi

$$(\sigma_Y^2)^{min} = \sigma_X^2 + 2(\underline{a}^{opt})^t\underline{r} - (\underline{a}^{opt})^t\underline{r} = \sigma_X^2 + (\underline{a}^{opt})^t\underline{r}. \tag{1.7}$$

On remarque que l'ensemble des deux équations (1.6) et (1.7) admet la représentation matricielle unique

$$\begin{bmatrix} r_X(0) & r_X(1) & \cdots & r_X(P) \\ r_X(1) & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & r_X(1) \\ r_X(P) & \cdots & r_X(1) & r_X(0) \end{bmatrix} \begin{bmatrix} 1 \\ a_1 \\ \vdots \\ a_P \end{bmatrix} = \begin{bmatrix} \sigma_Y^2 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{1.8}$$

Comparaison des deux approches

Les deux solutions sont très comparables ce qui n'est pas étonnant puisque $\frac{1}{N}\Gamma^t\underline{x}$ est une estimée du vecteur $\underline{r}$ et $\frac{1}{N}\Gamma^t\Gamma$ est une estimée de $\mathbb{R}$. Plus précisément les deux approches sont "asymptotiquement" équivalentes puisque, si le signal $X(n)$ est un p.a. "ergodique", c'est à dire si

$$\lim_{N\to\infty} \frac{1}{2N+1} \sum_{n=-N}^{+N} x(n)x(n+k) = \mathbb{E}\{X(n)X(n+k)\}$$

on a

$$\lim_{N\to\infty} \frac{\Gamma^t\Gamma}{N} = \mathbb{R} \quad \text{et} \quad \lim_{N\to\infty} \frac{\Gamma^t\underline{x}}{N} = \underline{r}.$$

La différence (essentielle mais subtile) est que la matrice de covariance exacte étant définie positive (sauf dans le cas limite où le processus est harmonique), cette matrice est toujours inversible alors que la matrice $\Gamma^t\Gamma$ (exprimée ici pour $P = 1$ pour simplifier)

$$\begin{bmatrix} x^2(1) + \cdots x^2(N-2) & x(0)x(1) + \cdots + x(N-3)x(N-2) \\ x(0)x(1) + \cdots + x(N-3)x(N-2) & x^2(0) + \cdots x^2(N-3) \end{bmatrix}$$

reste symétrique mais elle n'est plus forcément définie positive.

Dans la pratique où on ne dispose que de N données observées, on désire maintenir cette propriété de positivité. On peut observer qu'estimer la fonction d'autocovariance par

$$\hat{r}_X(k) = \frac{1}{N} \sum_{n=0}^{N-1-k} x(n)x(n+k) \quad \text{pour} \quad k = 0 \cdots P$$

puis construire $\hat{\mathbb{R}}$ et $\underline{\hat{r}}$ à partir de $\hat{r}_X(k)$ maintient le caractère défini positif de $\hat{\mathbb{R}}$ donc permet toujours son inversion. On peut alors déterminer les coefficients du filtre par

$$\underline{a}^{opt} = -\hat{\mathbb{R}}^{-1}\underline{\hat{r}} \tag{1.9}$$

et la puissance du signal "résiduel" par

$$\hat{\sigma}_Y^2 = \hat{r}_X(0) + (\underline{a}^{opt})^t \underline{\hat{r}}.$$

On dit que l'on fait "une analyse LPC" (Linear Predictive Coding).

On peut montrer aussi que la propriété de positivité entraîne que tous les zéros du polynôme $A(z)$ sont à l'intérieur du cercle unité ce qui assure la stabilité du filtre $1/A(z)$. C'est une propriété très importante dans la pratique comme on le verra par la suite lorsque l'on s'intéressera au codage de la parole à débit réduit.

Filtre "blanchissant"

On peut montrer que l'erreur de prédiction $Y(n)$ est blanche (plus exactement que le signal $X(n)$ a été blanchi). En effet, on remarque que

$$\frac{\partial E\{|Y(n)|^2\}}{\partial a_i} = 0 \quad \Rightarrow \quad E\{Y(n)X(n-i)\} = 0 \quad \forall i = 1 \cdots P.$$

Supposons P grand. Comme $Y(n)$ est non corrélé avec tous les $X(n-i)$ précédents et que $Y(n-i)$ est une combinaison linéaire de ces $X(n-i)$, on en déduit que $Y(n)$ est non corrélé avec $Y(n-i)$. L'erreur de prédiction $Y(n)$ est donc un bruit blanc mais cette propriété n'est vérifiée *a priori* que si $P \to \infty$ (comportement asymptotique). On appelle le filtre donnant $Y(n)$ à partir de $X(n)$ le "filtre blanchissant".

Si $Y(n)$ a été totalement blanchi, on peut écrire

$$S_Y(f) = |A(f)|^2 S_X(f) = \sigma_Y^2.$$

On a donc

$$S_X(f) = \frac{\sigma_Y^2}{|A(f)|^2}.$$

On rappelle que l'estimateur spectral le plus standard est le périodogramme : à partir des N données observées $[x(0) \ \cdots \ x(N-1)]$ on calcule

$$S_X(f_k = \frac{k}{N}) = \frac{1}{N}|\sum_{n=0}^{N-1} x(n)\exp(-j2\pi\frac{k}{N}n)|^2 \quad \text{pour} \quad k = 0 \cdots N/2.$$

La formule $S_X(f) = \sigma_Y^2/|A(f)|^2$ suggère un deuxième estimateur spectral : à partir des N données observées, on calcule les coefficients du filtre blanchissant en réalisant une analyse LPC puis on exploite la formule précédente.

Algorithme de Levinson

Pour calculer le prédicteur optimal à l'ordre P, il suffit de résoudre le système linéaire de P équations à P inconnues donné par (1.6) ou par (1.9). On peut, par exemple, utiliser l'algorithme de Gauss. Celui-ci nécessite $O(P^3)$ opérations. Il existe des algorithmes rapides réclamant $O(P^2)$ opérations, exploitant les propriétés de centro-symétrie de la matrice $\mathbb{R}$. Le fait qu'ils soient rapides ne présentent plus autant d'intérêt que dans les années 60 lorsqu'ils ont été conçus mais ils présentent toujours de l'intérêt, spécialement en codage de la parole, car ils mettent en évidence des paramètres équivalents aux coefficients a_i et présentant de meilleurs facultés de codage.

L'algorithme le plus célèbre est *l'algorithme de Levinson*. On donne la description de cet algorithme sans aucune justification. On consultera par exemple [4] pour plus de détails.

C'est un algorithme récursif sur l'ordre : connaissant le prédicteur optimal à l'ordre j, on obtient le prédicteur à l'ordre $j+1$. On note $a_1^j \cdots a_j^j$ les coefficients du prédicteur à l'ordre j, $\rho_j = r_X(j)/r_X(0)$ les coefficients d'autocovariance normalisés et σ_j^2 la variance de l'erreur de prédiction à cet ordre. Lorsque l'indice j atteint l'ordre P, on a $a_i^{j=P} = a_i$ et $\sigma_{j=P}^2 = \sigma_Y^2$.

- $a_1^1 = k_1 = -\rho_1$
- $\sigma_1^2 = (1-k_1^2)\sigma_X^2$
- Pour $j = 2 \cdots P$
  - $\delta_j = \rho_j + a_1^{j-1}\rho_{j-1} + \cdots + a_{j-1}^{j-1}\rho_1$
  - $k_j = -\delta_j/\sigma_{j-1}^2$
  - Pour $i = 1 \cdots j-1$: $a_i^j = a_i^{j-1} + k_j a_{j-i}^{j-1}$
  - $a_j^j = k_j$
  - $\sigma_j^2 = \sigma_{j-1}^2 + k_j\delta_j = \sigma_{j-1}^2(1-k_j^2)$.

Les coefficients $k_1 \cdots k_P$ sont appelés les coefficients de corrélation partielle (PARCOR). La dernière équation de l'algorithme précédent montre que tous ces coefficients sont en module inférieur à 1 puisque les variances sont toujours positives. C'est cette propriété qui les rend particulièrement intéressant en codage.

En écrivant sous forme matricielle l'ensemble des équations (1.8) pour $j = 0 \cdots P$ sans chercher à expliciter la partie triangulaire supérieure de la matrice apparaissant au second membre, on obtient[6]

$$\mathbb{R}\begin{bmatrix} 1 & 0 & \cdots & 0 \\ a_1^P & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ a_P^P & \cdots & a_1^1 & 1 \end{bmatrix} = \begin{bmatrix} \sigma_P^2 & x & \cdots & x \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & x \\ 0 & \cdots & 0 & \sigma_0^2 \end{bmatrix} \tag{1.10}$$

qui s'interprète comme une décomposition de Choleski de la matrice d'autocovariance.

1.3.3 Gain de prédiction

Définition

On appelle gain de prédiction (gain dû à la prédiction) le rapport des puissances des erreurs de quantification obtenues en utilisant le quantificateur optimal sans prédiction et avec prédiction à résolution b constante. Il est égal à

$$G_p = \frac{c\ \sigma_X^2\ 2^{-2b}}{c\ \sigma_Y^2\ 2^{-2b}} = \frac{\sigma_X^2}{\sigma_Y^2}. \tag{1.11}$$

[6] On observera que la matrice d'autocovariance $\mathbb{R}$ est cette fois ci de dimension $P+1$ alors qu'elle était de dimension P précédemment. On ne fait pas de distinction dans les notations pour ne pas les alourdir tant que cette distinction n'est pas vraiment nécessaire.

En exploitant (1.7), le gain de prédiction a pour expression

$$G_p(P) = \frac{r_X(0)}{r_X(0) + \sum_{i=1}^{P} a_i^{opt} r_X(i)}.$$

Il dépend de l'ordre de la prédiction P. Le gain de prédiction peut aussi s'écrire en fonction des coefficients PARCOR. Comme $\sigma_Y^2 = \sigma_X^2 \prod_{i=1}^{P}(1-k_i^2)$ le gain de prédiction est égal à

$$G_p(P) = \frac{1}{\prod_{i=1}^{P}(1-k_i^2)}.$$

C'est une fonction croissante de P. On peut montrer qu'il tend vers une limite $G_p(\infty)$ que l'on appellera la valeur asymptotique du gain de prédiction.

1.3.4 Valeur asymptotique du gain de prédiction

Cette valeur asymptotique peut s'exprimer sous différentes formes, par exemple en fonction du déterminant de la matrice d'autocovariance. En effet en prenant le déterminant des deux membres de l'équation (1.10), on obtient[7]

$$\det \mathbb{R}(P+1) = \prod_{j=0}^{P} \sigma_j^2.$$

Généralement lorsque l'on augmente l'ordre de prédiction, la puissance de l'erreur de prédiction décroit rapidement puis reste pratiquement constante à partir d'un certain ordre P_0. Cela est dû au fait que le signal n'étant plus corrélé au delà de cet ordre, on ne peut plus améliorer la prédiction en augmentant l'ordre. En appelant σ_Y^2 la plus petite puissance possible et pour P suffisamment élevé, on a

$$\det \mathbb{R}(P+1) \approx (\sigma_Y^2)^{P+1}.$$

On a donc

$$G_p(\infty) = \frac{\sigma_X^2}{\lim_{P\to\infty}[\det \mathbb{R}(P)]^{1/P}}. \tag{1.12}$$

La valeur asymptotique du gain de prédiction peut aussi s'exprimer en fonction de la densité spectrale de puissance $S_X(f)$ du p.a. $X(n)$.

Montrons d'abord que la puissance de l'erreur de prédiction en utilisant tout le passé d'un processus stationnaire de densité spectrale $S_X(f)$ est donnée par

$$\sigma_Y^2 = \exp\left(\int_{-1/2}^{1/2} \log_e S_X(f) df\right).$$

En effet, écrivons $B(z)$ sous la forme

$$B(z) = 1 + \sum_{i=1}^{P} a_i^{opt} z^{-i} = \prod_{i=1}^{P}(1 - p_i z^{-1}) \quad \text{avec} \quad |p_i| < 1$$

puisque le polynôme $B(z)$ est à déphasage minimal. On peut donc écrire

$$\log_e\left[\prod_{i=1}^{P}(1 - p_i z^{-1})\right] = \sum_{i=1}^{P} \log_e(1 - p_i z^{-1}) = \alpha_1 z^{-1} + \alpha_2 z^{-2} + \cdots$$

[7] en spécifiant explicitement dans cette section la dimension de la matrice $\mathbb{R}$

pour z appartenant à un domaine d'existence qui inclut le cercle unité. On a donc

$$\int_{-1/2}^{+1/2} \log_e[B(f)]df = 0$$

en appliquant le théorème de Cauchy. On en déduit

$$\int_{-1/2}^{+1/2} \log_e[S_X(f)]df = \int_{-1/2}^{+1/2} \log_e(\sigma_Y^2)df - \int_{-1/2}^{+1/2} \log_e(|B(f)|^2)df = \log_e(\sigma_Y^2)$$

ce qui entraîne le résultat.

Comme la puissance du signal est égale à

$$\sigma_X^2 = \int_{-1/2}^{1/2} S_X(f)df$$

on obtient une nouvelle expression pour la valeur asymptotique du gain de prédiction

$$\boxed{G_p(\infty) = \frac{\int_{-1/2}^{1/2} S_X(f)df}{\exp(\int_{-1/2}^{1/2} \log_e S_X(f)df)}} \tag{1.13}$$

comme une fonction de la densité spectrale de puissance du signal à quantifier uniquement.

Cette expression s'interprète comme le rapport entre la moyenne arithmétique et la moyenne géométrique de $S_X(f)$. En effet, si on considère $S_X(f)$ évalué sur N valeurs dans l'intervalle $[-1/2, 1/2]$ ou ce qui revient au même dans l'intervalle $[0, 1]$, on obtient

$$\int_0^1 S_X(f)df \approx \frac{1}{N}\sum_{k=0}^{N-1} S_X(\frac{k}{N})$$

$$\exp(\int_0^1 \log_e S_X(f)df) \approx \exp(\frac{1}{N}\sum_{k=0}^{N-1} \log_e S_X(k/N)) = \exp(\log_e(\prod_{k=0}^{N-1} S_X(k/N))^{1/N})$$

$$\exp(\int_0^1 \log_e S_X(f)df) \approx (\prod_{k=0}^{N-1} S_X(\frac{k}{N}))^{1/N}.$$

Le signal le moins prédictible est le bruit blanc. La valeur asymptotique du gain de prédiction est égale à 1 comme le montre la formule (1.13). La moyenne arithmétique et la moyenne géométrique sont égales. Aucun gain ne peut être espéré en utilisant une quantification scalaire prédictive plutôt qu'une quantification scalaire standard. Inversement le signal le plus prédictible est un processus de la forme

$$X(n) = \sum_{k=1}^{K} a_k \cos(2\pi f_k n + \Phi_k)$$

où Φ_k sont des phases aléatoires. Le gain de prédiction est infini. Comme la puissance de l'erreur de quantification, ou de l'erreur de reconstruction, a pour limite

$$\sigma_Q^2 = c\frac{\sigma_X^2}{G_p(\infty)}\ 2^{-2b}$$

on voit que l'on peut quantifier sans distorsion un processus harmonique quel que soit le choix de b. C'est évidemment purement théorique car cela veut simplement dire qu'il suffit de coder les différentes phases avec un nombre fini de bits et qu'ensuite plus aucune information n'a besoin d'être transmise pendant un temps aussi long que l'on veut !

Le rapport inverse de la valeur asymptotique du gain de prédiction porte le nom de mesure d'étalement spectral.

1.3.5 Quantification scalaire prédictive en boucle fermée

Reprenons le schéma de principe du quantificateur prédictif donné figure 1.3. Sous cette forme, le quantificateur exige la transmission à chaque instant n non seulement du numéro $i(n)$, résultat de la quantification de l'erreur de prédiction $y(n)$, mais aussi d'un autre numéro qui serait associé à la quantification de la prédiction $v(n)$ elle-même. Ce schéma de quantification, appelé schéma de quantification en boucle ouverte, n'est donc pas réaliste puisque l'on n'a pas intérêt, à résolution constante, à multiplier les informations à coder. On préfère réaliser une prédiction *en boucle fermée* ou *autour du quantificateur* comme le montre la figure 1.4 puisque l'on peut alors consacrer toutes les ressources binaires disponibles à la quantification de l'erreur de prédiction $y(n)$. Il n'est plus nécessaire de transmettre $v(n)$ au récepteur puisque $v(n)$ représente maintenant la prédiction du signal reconstruit $\hat{x}(n)$. Cette prédiction peut être réalisée de façon identique à l'émetteur. Il suffit qu'une copie du traitement réalisé au récepteur soit faite à l'émetteur. On parle de décodeur local (à l'émetteur) et de décodeur lointain (au récepteur). Cette façon de procéder a un coût : la prédiction se fait sur le signal reconstruit $\hat{x}(n)$ et non sur le signal original $x(n)$. Ce n'est pas grave tant que $\hat{x}(n)$ est une bonne approximation de $x(n)$, c'est-à-dire lorsque le taux de compression visé est peu élevé.

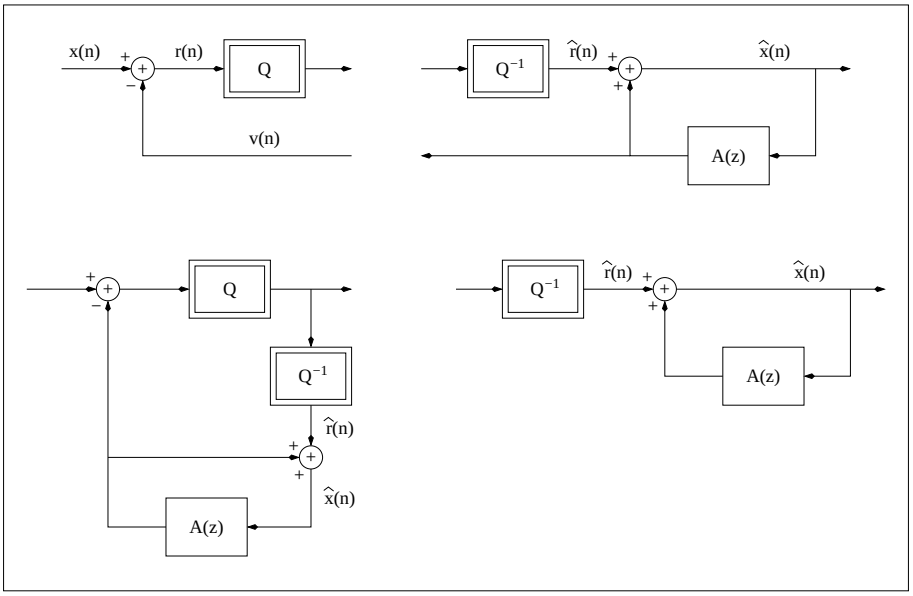

FIG. 1.4 – Quantificateur prédictif en boucle fermée.

On peut se poser maintenant le problème de la détermination des coefficients du polynôme $A(z)$. Les signaux que l'on cherche à quantifier n'étant pas stationnaires, il faut actualiser ces coefficients à intervalles réguliers. Si les signaux peuvent être considérés comme localement stationnaires sur N échantillons, il suffit d'actualiser ces coefficients tous les N échantillons. Le calcul se fait généralement à partir du signal $x(n)$. On dit que la prédiction est calculée de façon *forward*. Bien qu'il soit nécessaire de transmettre cette information au décodeur, ce transfert est possible car il peut réclamer un débit peu élevé. On peut aussi calculer les coefficients du filtre à partir du signal $\hat{x}(n)$. Dans ce cas, on dit que la prédiction est calculée de façon *backward*. Il n'est pas nécessaire alors de transmettre cette information. L'adaptation peut même se faire à l'arrivée de chaque nouvel échantillon $\hat{x}(n)$ par un algorithme du gradient (méthode adaptative).

Comparons plus en détail les avantages et les inconvénients respectifs de ces deux méthodes. La prédiction *forward* utilise des données plus fiables (c'est particulièrement important lorsque les propriétés statistiques du signal évoluent rapidement) mais il faut transmettre une *information*

adjacente (side-information) et il faut attendre l'arrivée du dernier échantillon de la fenêtre courante avant de commencer la procédure d'encodage sur l'ensemble de la fenêtre. On a donc un délai de reconstruction d'au moins N échantillons. Avec une prédiction *backward* le délai de reconstruction peut être rendu très court mais la prédiction est moins bonne car réalisée à partir d'échantillons dégradés. On remarque également que, dans ce cas, le codeur est plus sensible aux erreurs de transmission. Ce choix est essentiellement fonction de l'application.

Il reste à examiner le problème de la stabilité du filtre au décodeur puisque il est autorégressif. On ne peut, en aucun cas, accepter un risque d'instabilité. On a vu que, si l'estimation de la matrice d'autocovariance est faite en maintenant son caractère défini positif, alors la stabilité du filtre est assurée, les pôles de la fonction de transfert sont à l'intérieur du cercle unité.

Chapitre 2

Quantification vectorielle

2.1 Introduction

Lorsque la résolution est faible, il est naturel de vouloir regrouper plusieurs échantillons $x(n)$ dans un vecteur $\underline{x}(m)$ et de chercher à quantifier l'ensemble. On parle de quantification vectorielle. Entre la résolution b, la dimension N des vecteurs et la taille L du dictionnaire, on a alors la relation

$$L = 2^{bN}.$$

Il n'est plus nécessaire que b soit un entier. Il suffit que le produit bN le soit ou même que, simplement, L le soit. La quantification vectorielle permet donc de définir des résolutions fractionnaires. Mais ce n'est pas la propriété essentielle : le quantificateur vectoriel permet de prendre en compte directement la corrélation contenue dans le signal plutôt que de chercher d'abord à décorréler le signal puis à quantifier un signal décorrélé comme le fait le quantificateur scalaire prédictif. Ce quantificateur serait parfait s'il n'avait pas un défaut majeur : la complexité du traitement en termes du nombre de multiplications/additions à traiter est exponentiel en fonction de N.

2.2 Formalisme

La quantification vectorielle est une généralisation immédiate de la quantification scalaire. On appelle quantificateur vectoriel de dimension N et de taille L une application de R^N dans un ensemble fini C contenant L vecteurs de dimension N

$$Q : \ R^N \longrightarrow C \quad \text{avec} \quad C = \{\underline{\hat{x}}^1 \cdots \underline{\hat{x}}^L\} \quad \text{où} \quad \underline{\hat{x}}^i \in R^N.$$

L'espace R^N est partitionné en L régions ou cellules définies par

$$\Theta^i = \{\underline{x} : Q(\underline{x}) = \underline{\hat{x}}^i\}.$$

On appelle C un dictionnaire, assimilable à une matrice si nécessaire, et $\underline{\hat{x}}^i$ un représentant, un vecteur de sortie ou un vecteur de reproduction. On dit également que C représente l'alphabet de reproduction et $\underline{\hat{x}}^i$ les symboles de reproduction conformément au vocabulaire habituel en théorie de l'information.

La formule (1.1) se généralise directement en utilisant la densité de probabilité conjointe et la norme euclidienne

$$\sigma_Q^2 = \frac{1}{N} \sum_{i=1}^{L} \int_{\underline{u} \in \Theta^i} ||\underline{u} - \underline{\hat{x}}^i||^2 p_X(\underline{u}) d\underline{u}. \tag{2.1}$$

Il s'agit de déterminer le dictionnaire C, c'est à dire les vecteurs de reproduction $\{\underline{\hat{x}}^1 \cdots \underline{\hat{x}}^L\}$ minimisant σ_Q^2. Cette minimisation conduit à deux conditions nécessaires d'optimalité qui s'expriment de façon identique au cas scalaire.

1. Etant donné un dictionnaire $C = \{\underline{\hat{x}}^1 \cdots \underline{\hat{x}}^L\}$, la meilleure partition est celle qui vérifie

$$\Theta^i = \{\underline{x} : ||\underline{x} - \underline{\hat{x}}^i||^2 \leq ||\underline{x} - \underline{\hat{x}}^j||^2 \quad \forall j \in \{1 \cdots L\}\ \}.$$

   C'est la règle du plus proche voisin. Cette partition est appelée la partition de Voronoï.

2. Etant donné une partition, les meilleurs représentants sont obtenus par la condition du centroïde. Pour une distorsion quadratique

$$\hat{x}^i = \mathbb{E}\{\underline{X}|\underline{X} \in \Theta^i\} = \frac{\int_{\Theta^i} \underline{u} p_X(\underline{u}) d\underline{u}}{\int_{\Theta^i} p_X(\underline{u}) d\underline{u}}.$$

On montre sur le tracé de gauche de la figure 2.1 une réalisation $x(n)$ d'un p.a. AR(2) gaussien et centré en fonction de n. Le tracé de droite est obtenu à partir de cette réalisation en formant des vecteurs de dimension $N = 2$ soit $\underline{x}(m) = [x(2m)\ x(2m+1)]^t$ et en projetant chaque vecteur $\underline{x}(m)$ dans le plan. On obtient un nuage de points qui a tendance à s'aligner d'autant plus le long de la première diagonale que le premier coefficient d'autocovariance normalisé se rapproche de 1.

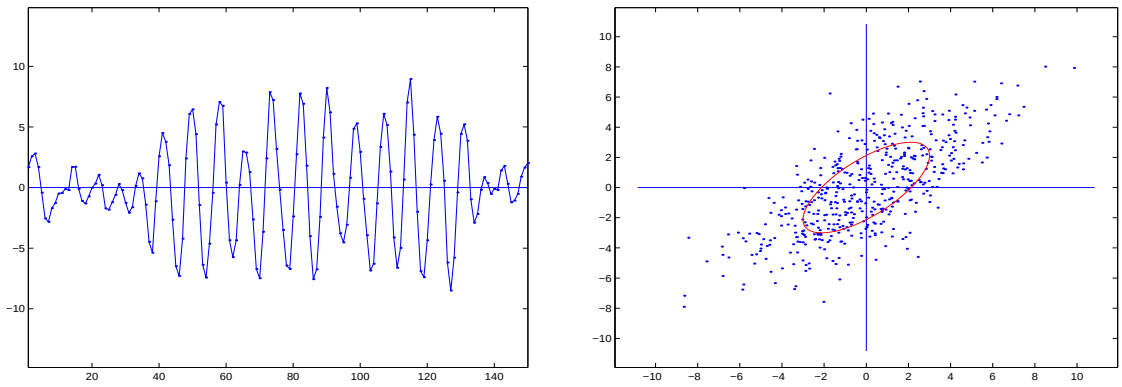

FIG. 2.1 – Exemple d'une réalisation d'un p.a. AR(2).

Les tracés de la figure 2.2 montrent deux façons de partitionner le plan. La partition de Voronoï correspondant au quantificateur vectoriel a été obtenue par application de l'algorithme de Lloyd-Max généralisé (cf section suivante) au cas vectoriel lorsque $b = 2$ et $N = 2$ c'est à dire lorsque le nombre de représentants L est égal à 16. La partition correspondant au quantificateur scalaire, interprétée dans le plan, impose des éléments de forme rectangulaire et des positions pour les représentants identiques relativement aux deux axes. On peut montrer que le rapport des deux axes de l'ellipse sur le deuxième tracé de la figure 2.1 est égal à $(1+\rho_1)/(1-\rho_1)$ où ρ_1 est le coefficient de covariance d'ordre 1 normalisé. On en déduit que, plus la corrélation est forte entre les composantes du vecteur, plus est efficace le quantificateur vectoriel parce qu'il s'adapte à la configuration du nuage de points tandis que le quantificateur scalaire ne subit pratiquement pas de modification. Le quantificateur vectoriel permet de prendre en compte directement la corrélation contenue dans le signal plutôt que de chercher d'abord à décorréler le signal puis à quantifier un signal décorrélé comme le fait le quantificateur scalaire prédictif.

La figure 2.3 représente le cas d'une sinusoïde entachée de bruit. On voit clairement que le quantificateur vectoriel s'adapte beaucoup mieux aux caractéristiques du signal.

Un théorème dû à Shannon [5] montre même que pour des signaux non-corrélés, des sources sans mémoire pour employer la terminologie habituelle en théorie de l'information, il y a un gain réalisable par quantification vectorielle. Ce problème présente de fortes analogies avec celui de l'empilement de sphères [6].

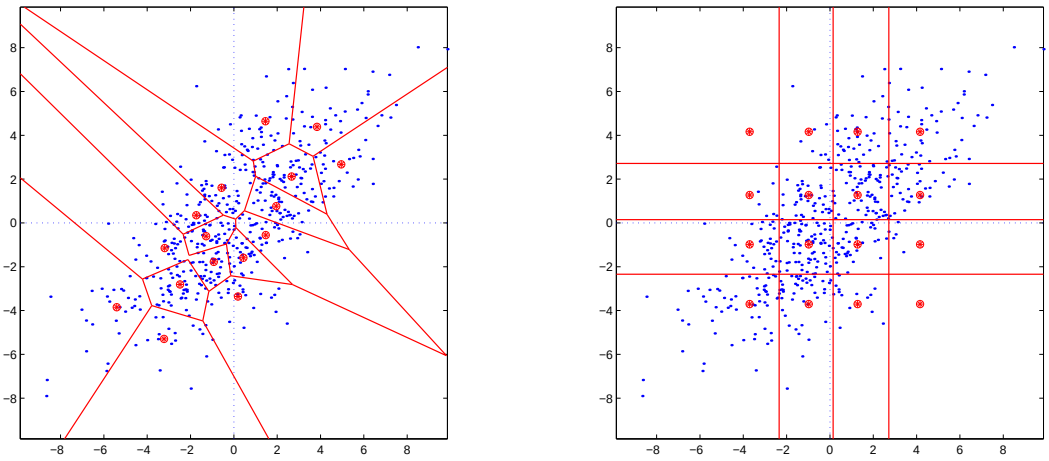

FIG. 2.2 – Comparaison des performances entre quantification vectorielle et quantification scalaire lorsque la résolution est égale à 2. Le quantificateur vectoriel comprend $L = 16$ représentants de dimension 2. Le quantificateur scalaire comprend $L = 4$ représentants.

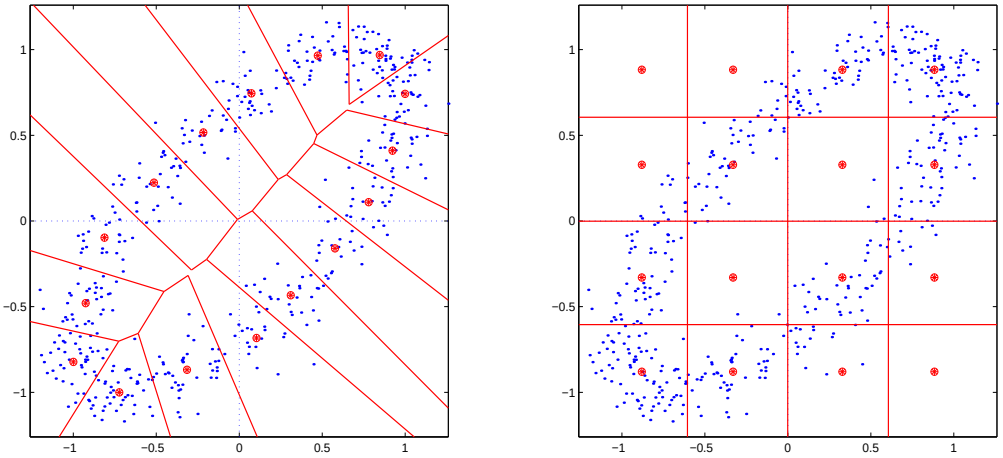

FIG. 2.3 – Comparaison des performances entre quantification vectorielle et quantification scalaire pour une sinusoïde entachée de bruit.

2.3 Construction du dictionnaire optimal

Dans la pratique, on ne connaît pas la densité de probabilité conjointe $p_X(\underline{x})$. Pour construire un quantificateur, on utilise des données empiriques, une base d'apprentissage, en associant à chaque valeur le même poids. La base d'apprentissage doit être composée d'un grand nombre d'échantillons représentatifs de la source. Pour constituer une base d'apprentissage caractéristique du signal de parole par exemple, on utilise plusieurs phrases phonétiquement équilibrées prononcées par plusieurs locuteurs masculins, féminins, jeunes, plus âgés, etc.

On donne ici une description sommaire de l'algorithme dit de Lloyd-Max permettant de construire un quantificateur. C'est un algorithme itératif vérifiant successivement les deux conditions d'optimalité.

1. On initialise le dictionnaire $\{\hat{\underline{x}}^1 \cdots \hat{\underline{x}}^L\}$, par exemple par tirage aléatoire.
2. Connaissant ce dictionnaire $\{\hat{\underline{x}}^1 \cdots \hat{\underline{x}}^L\}$, on étiquette chaque échantillon de la base d'apprentissage, par le numéro de son plus proche voisin. On détermine ainsi implicitement (le

calcul explicite n'est pas nécessaire) la partition optimale $\{\Theta^1 \cdots \Theta^L\}$.

3. A partir de tous les échantillons étiquetés par le même numéro, on en déduit un nouveau représentant par un calcul de moyenne.
4. On calcule la distorsion moyenne associée à cette base d'apprentissage et on arrête cet algorithme si la distorsion ne décroît presque plus, c'est-à-dire si la décroissance devient inférieure à un seuil, sinon on reprend les deux étapes précédentes.

On assure la décroissance de la distorsion moyenne mais on ne tend pas toujours vers le minimum global. On atteint simplement un minimum local. En fait, il n'existe même pas de théorèmes prouvant que l'on atteint un minimum local. De nouveaux algorithmes basés sur des techniques de recuit simulé, par exemple, permettent (en théorie) d'améliorer les performances du quantificateur.

L'initialisation du dictionnaire pose un problème. L'algorithme dit LBG [7], généralement adopté, permet de résoudre ce problème. Les différentes étapes sont les suivantes.

1. On cherche d'abord le dictionnaire composé d'un seul vecteur minimisant la distorsion moyenne. C'est le centre de gravité de l'ensemble de la base d'apprentissage. On le notera $\underline{\hat{x}}^0(b=0)$. Si on appelle L' le nombre de vecteurs composant la base d'apprentissage, la distorsion est égale à
$$\sigma_Q^2(b=0) = \frac{1}{L'}\frac{1}{N}\sum_{m=0}^{L'-1} ||\underline{x}(m)||^2 = \sigma_X^2$$
puisque le signal est supposé centré.
2. On *partage* ensuite ce vecteur en 2 vecteurs notés $\underline{\hat{x}}^0(b=1)$ et $\underline{\hat{x}}^1(b=1)$ avec $\underline{\hat{x}}^0(b=1) = \underline{\hat{x}}^0(b=0)$ et $\underline{\hat{x}}^1(b=1) = \underline{\hat{x}}^0(b=0) + \underline{\epsilon}$. Le choix du vecteur $\underline{\epsilon}$ pose un problème. On choisit des "petites" valeurs.
3. Connaissant $\underline{\hat{x}}^0(b=1)$ et $\underline{\hat{x}}^1(b=1)$, on classe tous les vecteurs de la base d'apprentissage relativement à ces deux vecteurs (on étiquette tous les vecteurs soit par 0 soit par 1) puis on calcule les nouveaux centres de gravité $\underline{\hat{x}}^0(b=1)$ et $\underline{\hat{x}}^1(b=1)$ de tous les vecteurs étiquetés respectivement 0 ou 1.
4. On calcule la distorsion
$$\sigma_Q^2(b=1) = \frac{1}{L'}\frac{1}{N}\sum_{m=0}^{L'-1} ||\underline{x}(m) - \underline{\hat{x}}||^2$$
et on itère ce processus tant que la décroissance de la distorsion reste importante. On remarquera que l'on assure constamment la décroissance grâce au choix particulier des initialisations. On applique ainsi un certain nombre de fois la règle du plus proche voisin et la condition du centroïde pour obtenir 2 vecteurs de reproduction minimisant la distorsion moyenne.
5. On partage à nouveau ces 2 vecteurs en 2 ...
6. On arrête l'algorithme lorsque l'on a atteint le nombre de vecteurs désiré.

2.4 Performances du quantificateur optimal

Dans le cadre de l'hypothèse de haute-résolution, Zador [8] a montré que la formule de Benett (cas scalaire)
$$\sigma_Q^2 = \frac{1}{12}\left(\int_R [p_X(x)]^{1/3} dx\right)^3 2^{-2b}$$

donnant la puissance de l'erreur de quantification en fonction de la densité de probabilité marginale du processus et de la résolution, se généralise au cas vectoriel. On obtient

$$\sigma_Q^2(N) = \alpha(N) \left(\int_{R^N} [p_X(\underline{x})]^{N/(N+2)} d\underline{x} \right)^{(N+2)/N} 2^{-2b}$$

où $p_X(\underline{x})$ représente maintenant la densité de probabilité conjointe et où $\alpha(N)$ est une constante qui ne dépend que de N. Dans le cas gaussien

$$\sigma_Q^2(N) = \alpha(N) \left(\int_{R^N} [\frac{1}{(2\pi)^{N/2}\sqrt{\det \mathbb{R}}} \exp(-\frac{1}{2}\underline{x}^t \mathbb{R}^{-1} \underline{x})]^{N/(N+2)} d\underline{x} \right)^{(N+2)/N} 2^{-2b}$$

$$\sigma_Q^2(N) = \alpha(N) \left((2\pi)^{N/(N+2)} (\frac{N+2}{N})^{N/2} (\det \mathbb{R})^{1/(N+2)} \right)^{(N+2)/N} 2^{-2b}$$

$$\sigma_Q^2(N) = \alpha(N) 2\pi (\frac{N+2}{N})^{(N+2)/2} (\det \mathbb{R})^{1/N} \; 2^{-2b}$$

soit

$$\sigma_Q^2(N) = c(N) (\det \mathbb{R})^{1/N} \; 2^{-2b}.$$

On montre que

$$c(1) = \frac{\sqrt{3}}{2}\pi > c(N) > c(\infty) = 1.$$

Lorsque $N = 1$, on retrouve la formule (1.5).

Comme dans le cas de la quantification scalaire prédictive, on peut chiffrer l'amélioration des performances apportée par la quantification vectorielle relativement à la quantification scalaire. Le *gain de quantification vectorielle* défini de façon similaire à (1.11) se décompose en deux termes

$$G_v(N) = \frac{c(1)}{c(N)} \times \frac{\sigma_X^2}{(\det \mathbb{R})^{1/N}}.$$

Le rapport $c(1)/c(N)$ est toujours supérieur à 1 ce qui montre que, même pour une source sans mémoire, une quantification vectorielle est préférable mais cette contribution reste limitée parce que

$$10 \log_{10} \frac{c(1)}{c(N)} < 10 \log_{10} \frac{c(1)}{c(\infty)} = 4.35 \; dB.$$

Le deuxième rapport traduit la prise en compte de la corrélation existant entre les différentes composantes du vecteur par le quantificateur vectoriel. Lorsque $N \to \infty$, il tend vers la valeur asymptotique du gain de prédiction $G_p(\infty)$ comme le montre l'équation (1.12).

La figure 2.4 montre les rapports signal sur bruit correspondant à la quantification vectorielle (en fonction de N) et à la quantification scalaire prédictive (en fonction de $P+1$) pour $b = 2$. On visualise également la limite du rapport signal sur bruit correspondant à la quantification vectorielle lorsque N tend vers l'infini. Le rapport signal sur bruit correspondant au quantificateur scalaire prédictif est égal à

$$RSB_{QSP} = 6,02 \; b - 4,35 + 10 \log_{10} G_p(\infty)$$

dès que $P \geq 2$. Le décalage de 4,35 dB existant entre les deux droites horizontales est dû au rapport $c(1)/c(\infty)$. Il chiffre le gain apporté par la grande souplesse du quantificateur vectoriel dans le choix de la forme géométrique de la partition.

Le quantificateur vectoriel exploite directement la corrélation existant dans le signal. Le quantificateur scalaire prédictif exploite également cette corrélation mais en réalisant au préalable une opération de décorrélation.

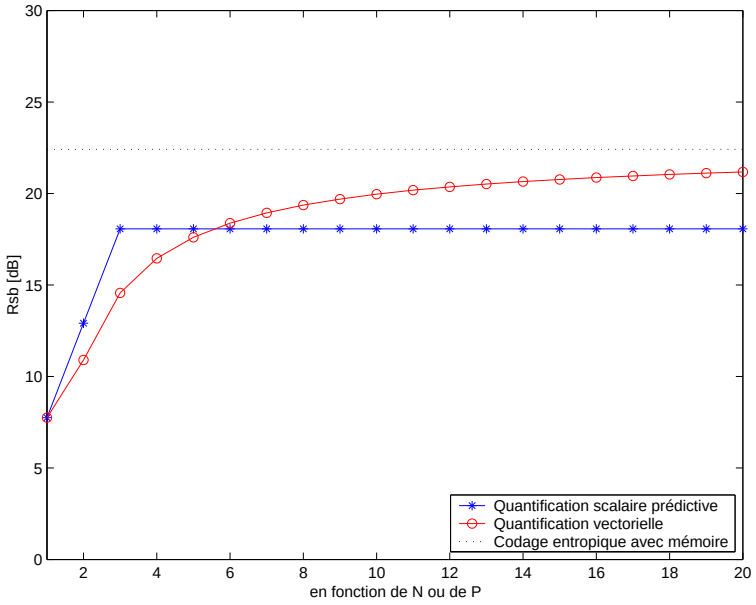

FIG. 2.4 – Rapport signal sur bruit en fonction de N pour le quantificateur vectoriel et en fonction de $P+1$ pour le quantificateur scalaire prédictif.

Dès que N devient supérieur à une valeur relativement faible, le quantificateur vectoriel a des performances meilleures que le quantificateur scalaire prédictif. Lorsque N augmente, les performances du quantificateur vectoriel se rapprochent rapidement de la valeur limite pour un processus stationnaire. On peut montrer qu'il n'existe pas de quantificateur procurant un rapport signal sur bruit meilleur que cette limite. Le quantificateur vectoriel peut donc être considéré comme le quantificateur optimal pourvu que N soit suffisamment élevé.

2.5 Utilisation du quantificateur

En principe, il suffit de regrouper les échantillons du signal à comprimer en une succession de vecteurs de dimension N, d'appliquer la règle du plus proche voisin pour obtenir le numéro du vecteur à l'encodage et d'extraire un vecteur dans une table à une adresse donnée pour fournir le vecteur de reproduction au décodage. Dans la pratique, toute une série de difficultés apparaissent, essentiellement dues à la puissance de calcul finie des processeurs car le traitement, aussi bien à l'encodage qu'au décodage, doit être généralement réalisé en temps réel. En fait, les calculs les plus lourds proviennent à l'encodage car au décodage il suffit d'aller chercher un vecteur dans une table à une adresse donnée.

Prenons l'exemple du codage du signal de parole en bande téléphonique à une résolution $b = 1$. On cherche, par exemple, à réaliser un codeur à 8 kbit/s. Comment choisir la dimension N des vecteurs et le nombre de vecteurs L composant le dictionnaire ? On vient de voir que l'on a intérêt à augmenter N mais la taille du dictionnaire croît alors de façon exponentielle puisque $L = 2^{bN}$, la charge de calcul (nombre de multiplications - accumulations) également puisqu'elle est proportionnelle à $NL = N2^{bN}$, soit 2^{bN} par échantillon ou $2^{bN} f_e$ multiplications - accumulations par seconde. On peut admettre que les microprocesseurs de signal actuels acceptent une charge de calcul de l'ordre de 10^8 multiplications - accumulations par seconde. On doit donc avoir

$$2^{bN} \times 8 \times 10^3 \leq 10^8$$

ce qui entraîne $N \leq 13$ pour $b = 1$. C'est très insuffisant pour au moins deux raisons. D'une part,

le signal de parole est trop complexe pour pouvoir être résumé à 2^{13} vecteurs. D'autre part, la fonction d'autocorrélation ne s'annule pas en quelques dizaines d'échantillons. Un quantificateur vectoriel ne réclame pas que les composantes du vecteur à quantifier soient décorrélées (corrélation intra-fenêtre) car il s'adapte justement à cette corrélation. Par contre, il faut chercher à décorréler au maximum les vecteurs entre eux (corrélation inter-fenêtre). Avec $N = 13$, les corrélations inter-fenêtres restent encore significatives pour du signal de parole.

Il est possible d'augmenter N et L sans trop modifier la charge de calcul en structurant le dictionnaire. De très nombreuses propositions ont été faites dont on trouvera un exposé très détaillé dans [3]. On ne donne ici qu'une description très sommaire des nombreuses possibilités.

2.5.1 Quantification vectorielle arborescente

On a une collection de dictionnaires composés chacun de deux vecteurs (cas d'un arbre binaire). A la première étape, on sélectionne le plus proche voisin de $\underline{x}(m)$ dans le premier dictionnaire. A la deuxième étape, on sélectionne le plus proche voisin de $\underline{x}(m)$ dans l'un ou l'autre des deux dictionnaires dépendant du premier choix, ... Le coût calcul par échantillon pour un dictionnaire standard est égal à 2^{bN}. Il devient égal à $2bN$ en imposant une structure arborescente binaire. Pour construire ce dictionnaire, l'algorithme LBG est spécialement adapté puisqu'il fournit tous les dictionnaires intermédiaires (après une légère adaptation). Remarquons qu'au décodage, seul le dictionnaire de taille 2^{bN} est utilisé.

2.5.2 Quantification vectorielle par produit cartésien

On décompose un vecteur en plusieurs sous-vecteurs de dimensions éventuellement différentes et on applique une quantification vectorielle particulière pour chaque sous-vecteur. On perd alors la possibilité d'exploiter la dépendance statistique qui peut exister entre toutes les composantes du vecteur. Il faut trouver une façon de décomposer un vecteur de telle sorte qu'il n'existe plus de dépendance entre les sous-parties. La complexité devient proportionnelle à $\sum_i L_i$ où L_i est le nombre de vecteurs composant le dictionnaire C_i.

2.5.3 Quantification vectorielle de type gain-forme

C'est une forme particulière du cas précédent. L'idée est de séparer dans deux dictionnaires distincts ce qui est caractéristique de la forme, le contenu spectral du vecteur, et ce qui est caractéristique du gain, l'énergie du vecteur. Le dictionnaire vectoriel est alors composé de vecteurs (éventuellement) normés. Prenons l'exemple du signal de parole. Que l'on parle doucement ou à voix forte, le contenu spectral du signal est peu modifié. Par contre, son énergie est fortement amplifiée. Ces deux types d'information sont fortement décorrélés ce qui justifie cette méthode.

2.5.4 Quantification vectorielle multi-étages

On procède par approximations successives. On réalise une première quantification vectorielle grossière, puis une deuxième quantification vectorielle portant sur l'erreur de quantification, etc. comme illustré figure 2.5.

2.5.5 Quantification vectorielle par transformée

On réalise une transformation orthogonale initiale puis on applique une quantification vectorielle sur le vecteur transformé. Puisqu'une transformation orthogonale laisse invariante la norme des vecteurs, l'erreur quadratique moyenne reste inchangée. *A priori*, cette transformation est peu intéressante. Pour coder des signaux, on réalise souvent une transformation initiale suivie

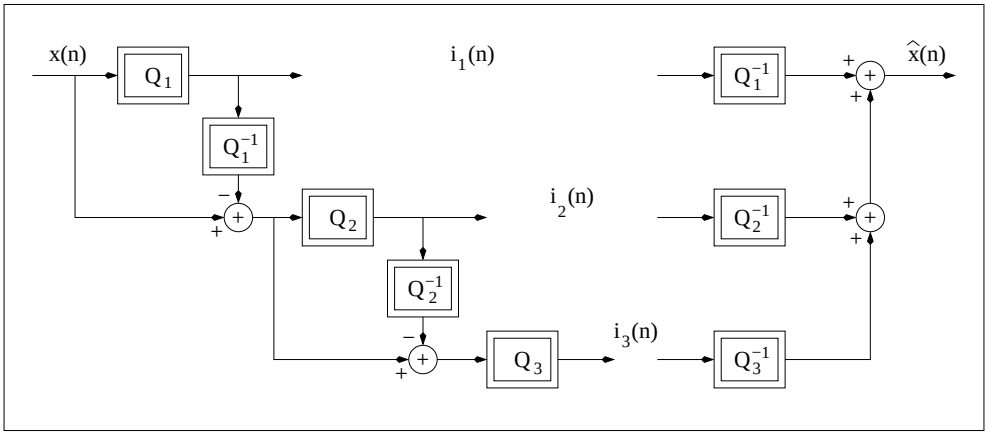

FIG. 2.5 – Quantification scalaire ou vectorielle multi-étages.

d'une quantification dans le domaine transformée mais cette quantification est généralement scalaire. Un chapitre entier est consacré à cette étude. Il est parfois utile d'associer transformation et quantification vectorielle pour pouvoir concentrer l'information dans un vecteur de dimension inférieure en négligeant les composantes très peu énergétiques. La réduction de la complexité peut être importante par cette opération de troncation. Des transformations orthogonales très classiques en Traitement du Signal sont par exemple la transformée de Fourier discrète et la transformée en cosinus discrète qui présente l'avantage, par rapport à la précédente, d'être à valeurs réelles.

2.5.6 Quantification vectorielle algébrique

Le dictionnaire n'est plus construit par application de l'algorithme de Lloyd-Max. Il est indépendant des propriétés statistiques de la source. Il consiste à répartir les vecteurs de reproduction de façon régulière dans l'espace. On parle alors de quantification vectorielle sur réseaux. Ces dictionnaires sont largement utilisés car ils permettent de réduire très fortement la charge de calcul et ils n'ont pas besoin d'être mémorisés.

2.6 Quantification vectorielle de type gain-forme

Une quantification vectorielle particulière, la quantification vectorielle "gain-forme", est largement utilisée, spécialement en codage audio, car cette quantification vectorielle permet de gérer commodément l'évolution au cours du temps de la puissance instantanée d'une source sonore. Plutôt que de minimiser la norme $||\underline{x}-\underline{\hat{x}}^j||$ relativement à j, on minimise $||\underline{x}-\hat{g}^i\underline{\hat{x}}^j||$ relativement à i et j. On montre, dans cette section, comment est modifiée la règle du plus proche voisin et comment on construit le dictionnaire optimal.

2.6.1 Règle du plus proche voisin

On utilise deux dictionnaires comme le montre la figure 2.6. Le premier $\{\underline{\hat{x}}^1 \cdots \underline{\hat{x}}^{L_1}\}$ est composé de L_1 vecteurs de dimension N éventuellement normés, le second $\{\hat{g}^1 \cdots \hat{g}^{L_2}\}$ est composé de L_2 scalaires. La minimisation de l'erreur quadratique relativement aux deux indices i et j peut être réalisée par une recherche exhaustive. On préfère, au prix d'une approximation, réaliser cette minimisation en deux temps. On recherche d'abord parmi tous les vecteurs du premier dictionnaire, le vecteur présentant la forme la plus proche du vecteur $\underline{x}$. On quantifie ensuite le gain de façon scalaire.

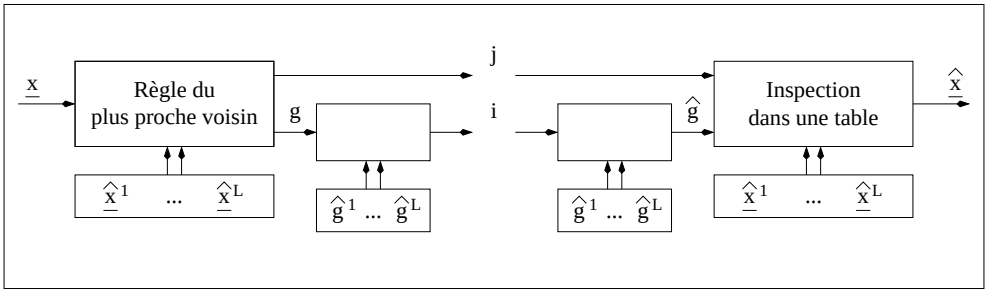

FIG. 2.6 – Quantificateur vectoriel de type forme-gain.

Considérons le schéma représenté figure 2.7. On appelle $g(j)$ le gain tel que le vecteur $g(j)\ \underline{\hat{x}}^j$

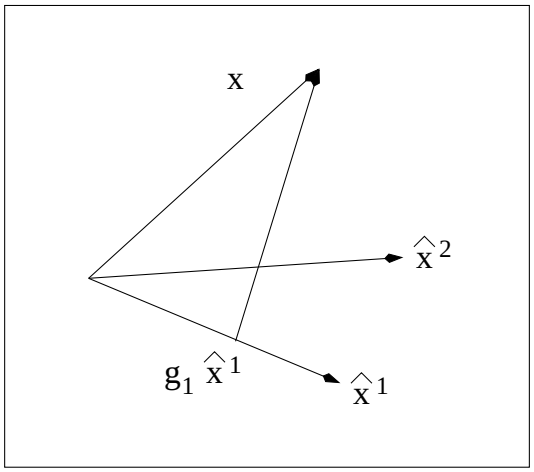

FIG. 2.7 – Quantification vectorielle forme-gain.

soit la projection orthogonale de $\underline{x}$ sur $\underline{\hat{x}}^j$. Ce gain est donné par

$$< \underline{x} - g(j)\underline{\hat{x}}^j, \underline{\hat{x}}^j >= 0$$

$$g(j) = \frac{< \underline{x}, \underline{\hat{x}}^j >}{||\underline{\hat{x}}^j||^2}$$

en appelant $< \underline{x}, \underline{y} >$ le produit scalaire de deux vecteurs $\underline{x}$ et $\underline{y}$. Comme

$$\min_j ||\underline{x} - g(j)\underline{\hat{x}}^j||^2 \equiv \min_j(||\underline{x}||^2 - 2g(j) < \underline{x}, \underline{\hat{x}}^j > +g^2(j)||\underline{\hat{x}}^j||^2)$$

$$\min_j ||\underline{x} - g(j)\underline{\hat{x}}^j||^2 \equiv \min_j(||\underline{x}||^2 - \frac{< \underline{x}, \underline{\hat{x}}^j >^2}{||\underline{\hat{x}}^j||^2})$$

en remplaçant $g(j)$ par sa valeur, la minimisation relativement à j devient une maximisation

$$\min_j ||\underline{x} - g(j)\underline{\hat{x}}^j||^2 \equiv \max_j \frac{< \underline{x}, \underline{\hat{x}}^j >^2}{||\underline{\hat{x}}^j||^2} \equiv \max_j < \underline{x}, \frac{\underline{\hat{x}}^j}{||\underline{\hat{x}}^j||} >^2 \equiv \max_j |cos\phi^j|$$

où ϕ^j représente l'angle compris entre $\underline{x}$ et $\underline{\hat{x}}^j$. La règle du plus proche voisin consiste donc, dans ce cas, à choisir parmi tous les vecteurs possibles, le vecteur présentant l'angle minimum (à π près) avec $\underline{x}$.

2.6.2 Algorithme de Lloyd-Max

Montrons comment on construit le premier dictionnaire, celui qui est caractéristique de la forme. On a vu que l'algorithme de Lloyd-Max se décompose en deux étapes. Connaissant une base d'apprentissage composé de M vecteurs $\underline{x}(m)$ et un dictionnaire initial composé des L_1 vecteurs $\underline{\hat{x}}^j$, que l'on supposera normés sans perte de généralité, on cherche d'abord la meilleure partition des M vecteurs en L_1 classes. On étiquette chaque vecteur de la base d'apprentissage par l'indice j qui maximise $< \underline{x}(m), \underline{\hat{x}}^j >^2$. Connaissant cette partition, on actualise ensuite le dictionnaire. A partir des $M(j)$ vecteurs $\underline{x}(m)$ associés à l'ancien vecteur $\underline{\hat{x}}^j$, on calcule le nouveau vecteur $\underline{\hat{x}}^j$ qui minimise l'erreur quadratique moyenne

$$\sigma_Q^2(j) = \frac{1}{M(j)}[\sum_{m=1}^{M(j)} ||\underline{x}(m)||^2 - \sum_{m=1}^{M(j)} < \underline{x}(m), \underline{\hat{x}}^j >^2].$$

Il suffit de maximiser par rapport à $\underline{\hat{x}}^j$ l'expression

$$Q = \sum_{m=1}^{M(j)} < \underline{x}(m), \underline{\hat{x}}^j >^2 = (\underline{\hat{x}}^j)^t \Gamma \underline{\hat{x}}^j$$

en appelant Γ la matrice de covariance empirique

$$\Gamma = \sum_{m=1}^{M(j)} \underline{x}(m)\underline{x}^t(m).$$

La maximisation de Q, sous la contrainte $||\underline{\hat{x}}^j|| = 1$, peut être réalisée par la méthode des multiplicateurs de Lagrange. On annule la dérivée par rapport à $\underline{\hat{x}}^j$ de l'expression

$$(\underline{\hat{x}}^j)^t \Gamma \underline{\hat{x}}^j - \lambda[(\underline{\hat{x}}^j)^t \underline{\hat{x}}^j - 1].$$

On obtient

$$\Gamma \underline{\hat{x}}^j = \lambda \underline{\hat{x}}^j.$$

Le paramètre λ est donc une valeur propre de la matrice Γ. En prémultipliant cette équation par $(\underline{\hat{x}}^j)^t$, on obtient

$$(\underline{\hat{x}}^j)^t \Gamma \underline{\hat{x}}^j = \lambda (\underline{\hat{x}}^j)^t \underline{\hat{x}}^j = \lambda.$$

Le premier terme étant l'expression que l'on cherche à maximiser, il suffit de choisir le vecteur propre associé à la plus grande valeur propre de la matrice de covariance empirique. C'est un résultat classique d'analyse de données en composantes principales [2].

Chapitre 3

Codage par transformée, codage en sous-bandes

3.1 Introduction

Jusqu'à une période récente, on faisait classiquement la distinction entre codage par transformée et codage en sous-bandes. Dans un codeur par transformée, on construit d'abord, à partir du signal, un vecteur de dimension N. On applique ensuite une transformée caractérisée par une matrice généralement orthogonale ou unitaire (par exemple construite à partir de la transformée de Fourier discrète ou de la transformée en cosinus discrète) puis on code les coefficients dans le domaine transformé. Au récepteur, on réalise la transformation inverse. Dans un codeur en sous-bandes, on commence par filtrer le signal par un banc de filtres d'analyse, on sous-échantillonne chaque signal de sous-bande puis on code séparément chaque signal sous-échantillonné. Au récepteur, on décode chaque composante, on réalise un sur-échantillonnage et une interpolation par un banc de filtres de synthèse puis on additionne les différents signaux reconstitués. Ces deux techniques de codage sont formellement équivalentes. On présentera cette équivalence dans la section 3.2. Pour plus détails, on consultera la littérature consacrée à la théorie des bancs de filtres à reconstruction parfaite [9, 10], aux systèmes multirésolution, aux transformées par ondelettes [11].

Dans la définition d'un système de codage, un problème important est l'allocation des bits disponibles entre différents sous-ensembles. On peut imaginer deux cas extrêmes : répartir des bits entre plusieurs parties hétérogènes, comme par exemple entre les coefficients d'un filtre et le signal d'excitation dans un codeur de parole, ou répartir des bits entre différents éléments présentant *a priori* la même distribution, comme par exemple les échantillons successifs d'un signal. On se propose de montrer, dans une deuxième partie, que les techniques de codage par transformée (ou en sous-bandes) consistent à chercher à rendre non-homogènes les différentes composantes d'un vecteur de façon à pouvoir réserver davantage de bits pour les composantes les plus significatives. On montrera que la transformation optimale, la transformée de Karhunen-Loeve, est celle qui décorrèle les composantes du vecteur transformé. Elle concentre le maximum de la puissance contenue dans le signal dans un nombre minimum de composantes.

On verra également que la minimisation de l'erreur quadratique moyenne, critère que l'on a privilégié jusqu'à présent, a pour effet de rendre le bruit de quantification le plus blanc possible. C'est une approche quelque peu théorique : en pratique, le problème n'est pas vraiment de chercher à rendre le bruit de quantification le moins puissant et le plus blanc possible. On cherche plutôt à imposer des contraintes à la forme spectrale du bruit pour respecter des contraintes psychoacoustqiues ou psychovisuelles. On parlera de codage "perceptuel". Cette question ne sera pas traitée dans ce chapitre. On abordera cette question dans la deuxième partie de ce livre.

3.2 Equivalence entre bancs de filtres et transformées

Considérons un banc de M filtres avec quantification (scalaire) des signaux de sous-bandes comme le montre le schéma de la figure 3.1. On suppose que tous les filtres sont des filtres

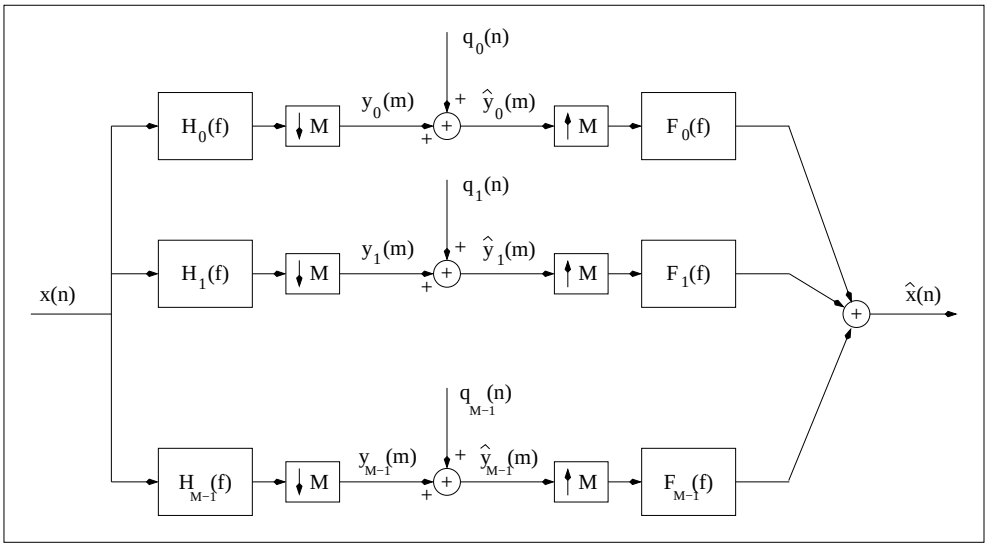

FIG. 3.1 – Schéma de principe d'un codeur en sous-bandes (avec sous-échantillonnage critique).

à réponse impulsionnelle finie, causaux et comportant le même nombre de coefficients N. On appelle M' le facteur de sous-échantillonnage et on note $b_0 \cdots b_{M-1}$ le nombre de bits que l'on accordera à la quantification de chaque signal de sous-bandes. On remarque que si $M' = M$, la quantification dans le domaine transformé porte sur le même nombre d'échantillons que si la quantification avait lieu dans le domaine temporel. Le sous-échantillonnage est alors "critique", le banc de filtres est "à décimation maximale". C'est le choix habituel en codage (que l'on supposera vérifié par la suite).

En construisant le vecteur $\underline{x}(m) = [x_0(m) \cdots x_{N-1}(m)]^t$ à partir de ses composantes polyphasées $x_l(m) = x(mM - l)$ et le vecteur $\underline{y}(m) = [y_0(m) \cdots y_{M-1}(m)]^t$ à partir des signaux de sous-bandes, les M équations

$$y_k(m) = \sum_{l=0}^{N-1} h_k(l) x(mM - l)$$

s'écrivent de façon matricielle

$$\underline{y}(m) = \begin{bmatrix} h_0(0) & \cdots & h_0(N-1) \\ \vdots & \ddots & \vdots \\ h_{M-1}(0) & \cdots & h_{M-1}(N-1) \end{bmatrix} \underline{x}(m) = \mathcal{T}\ \underline{x}(m). \tag{3.1}$$

Le banc de filtres d'analyse est donc équivalent à une transformation $\mathcal{T}$ caractérisée par une matrice rectangulaire $M \times N$ dont les vecteurs lignes sont les réponses impulsionnelles des filtres d'analyse. On montre de même que le banc de filtres de synthèse est équivalent à une transformation $\mathcal{P}$ caractérisée par une matrice rectangulaire $N \times M$ dont les vecteurs colonnes renversés sont les réponses impulsionnelles des filtres de synthèse. La condition générale de reconstruction parfaite, la condition de *bi-orthogonalité*, est simplement que le produit matriciel $\mathcal{P}\mathcal{T}$ donne la matrice identité lorsque $N = M$ (cas trivial). Dans le cas habituel lorsque $N > M$ (cas non-trivial), la condition de bi-orthogonalité se généralise [10].

Obtenir les réponses impulsionnelles $h_k(n)$ et $f_k(n)$ vérifiant cette condition est un problème difficile. Pour simplifier ce problème, on se contente presque toujours[1], de bancs de filtres "modulés" en introduisant une contrainte forte sur $h_k(n)$ et $f_k(n)$. Ces réponses impulsionnelles sont obtenues à partir des réponses impulsionnelles de filtres prototypes $h(n)$ et $f(n)$ par "modulation" : $h_k(n) = h(n)d_k(n)$ et $f_k(n) = f(n)d_k(n)$ pour $k = 0 \cdots M-1$ et $n = 0 \cdots N-1$. On remarque que l'équation (3.1) devient

$$\underline{y}(m) = D\ \underline{u}(m) \quad \text{avec} \quad \underline{u}(m) = \underline{h} \otimes\ \underline{x}(m)$$

où D est la "matrice de modulation" et où l'opération $\otimes$ consiste à multiplier deux vecteurs composantes par composantes. Le vecteur $\underline{u}(m)$ est simplement la version fenêtrée du vecteur $\underline{x}(m)$ par la fenêtre de pondération $\underline{h}$.

Il s'agit de déterminer $d_k(n)$, $h(n)$ et $f(n)$. Si on part d'un filtre prototype passe-bas idéal de fréquence de coupure $1/4M$ et si on construit à partir de ce filtre prototype les réponses en fréquence $H_0(f) \cdots H_{M-1}(f)$ des M filtres du banc de la façon suivante

$$H_k(f) = H(f - \frac{2k+1}{4M}) + H(f + \frac{2k+1}{4M}),$$

on en déduit directement la relation entre les réponses impulsionnelles

$$h_k(n) = 2\ h(n)\ cos(2\pi \frac{2k+1}{4M} n).$$

On obtient des relations équivalentes pour le banc de filtres de synthèse. Dans la pratique, il est impossible d'empêcher du recouvrement dans le domaine fréquentiel à cause du caractère fini de N. On montre qu'il existe tout de même des solutions au problème de la reconstruction parfaite. La plus classique est la transformée en cosinus discrète modifiée (MDCT, TDAC, MLT ...) où la matrice D est une matrice en cosinus avec des phases appropriées

$$d_k(n) = \sqrt{\frac{2}{M}} cos[(2k+1)(2n+1+N-M)\frac{\pi}{4M}] \quad \text{pour} \quad \begin{array}{rcl} k & = & 0 \cdots M-1 \\ n & = & 0 \cdots N-1. \end{array}$$

On montre que la condition de reconstruction parfaite s'écrit :

$$h^2(n) + h^2(n+M) = 1 \quad \text{pour} \quad n = 0 \cdots M-1 \tag{3.2}$$

avec

$$f(n) = h(n) \quad \text{pour} \quad n = 0 \cdots N-1$$

condition vérifiée par

$$h(n) = f(n) = sin[(2n+1)\frac{\pi}{4M}].$$

Cette transformation est très utilisée (en codage audio).

Il reste à déterminer N et M. C'est une question délicate dans la pratique car la transformation doit satisfaire des contraintes contradictoires. On désire généralement de bonnes propriétés fréquentielles (peu de recouvrement dans le domaine fréquentiel, résolution fréquentielle importante) sans trop nuire à la résolution temporelle (pour accepter des variations rapides de la puissance instantanée du signal). Cette question sera abordée dans la deuxième partie de ce livre.

[1] Exception : utilisation de transformées en ondelettes surtout en codage d'image.

3.3 Allocation de bits

3.3.1 Définition du problème

Les M composantes $y_k(m)$ du vecteur $\underline{y}(m) = \mathcal{T}\underline{x}(m)$ peuvent être interprétées comme la réalisation de M processus aléatoires stationnaires $Y_k(m)$ de puissance $\sigma^2_{Y_k}$. On note $\sigma^2_{Q_0} \cdots \sigma^2_{Q_{M-1}}$ les puissances des M erreurs de quantification. En admettant que la condition de reconstruction parfaite entraîne la conservation de la puissance comme dans le cas d'une transformation orthogonale ou unitaire (lorsque $N = M$), on en déduit que la puissance de l'erreur de reconstruction (celle qui intéresse l'utilisateur)

$$\sigma^2_Q = \mathbb{E}\{|X(n) - \hat{X}(n)|^2\} = \frac{1}{N}\mathbb{E}\{||\underline{X}(m) - \underline{\hat{X}}(m)||^2\}$$

est égale à la moyenne arithmétique des puissances des erreurs de quantification des différentes composantes (erreurs introduites localement par les quantificateurs)

$$\sigma^2_Q = \frac{1}{M}\mathbb{E}\{||\underline{Y}(m) - \underline{\hat{Y}}(m)||^2\} = \frac{1}{M}\sum_{k=0}^{M-1}\sigma^2_{Q_k}.$$

Le premier problème que l'on examine est le suivant. On applique une transformation $\mathcal{T}$ quelconque. On dispose de bM bits pour quantifier scalairement les M coefficients, les M signaux de sous-bandes. On cherche la façon optimale d'allouer les ressources binaires bM disponibles. Il s'agit donc de déterminer le vecteur $\underline{b} = [b_0 \cdots b_{M-1}]^t$ minimisant σ^2_Q sous la contrainte

$$\sum_{k=0}^{M-1} b_k \leq bM.$$

3.3.2 Allocation de bits optimale

Dans le cadre de l'hypothèse haute résolution, on peut exprimer $\sigma^2_{Q_k}$ en fonction de $\sigma^2_{Y_k}$ et de b_k en appliquant (1.5). On rappelle la formule en soulignant le fait que $\sigma^2_{Q_k}$ dépend de b_k

$$\sigma^2_{Q_k}(b_k) = c_k(1)\ \sigma^2_{Y_k}\ 2^{-2b_k} \tag{3.3}$$

où $c_k(1)$ est une constante qui ne dépend que de la densité de probabilité marginale de $Y_k(m)$. L'hypothèse de gaussiannité se répercutant sur les signaux de sous-bandes, toutes les constantes $c_k(1)$ sont égales. On appelle $c(1)$ cette constante commune.

Montrons que l'allocation optimale de bits minimisant

$$\sigma^2_Q = \frac{c(1)}{M}\sum_{k=0}^{M-1}\sigma^2_{Y_k}\ 2^{-2b_k} \tag{3.4}$$

sous la contrainte $\sum_{k=0}^{M-1} b_k \leq bM$ est donnée par

$$b_k = b + \frac{1}{2}\log_2\frac{\sigma^2_{Y_k}}{\alpha^2} \tag{3.5}$$

où b est le nombre moyen de bits par composante et où

$$\alpha^2 = (\prod_{k=0}^{M-1}\sigma^2_{Y_k})^{1/M}$$

est la moyenne géométrique des puissances.

En effet, on sait que, quels que soient M réels positifs, la moyenne arithmétique est supérieure ou égale à la moyenne géométrique et que l'égalité a lieu lorsque les M nombres sont égaux ce qui s'écrit

$$\frac{1}{M}\sum_{k=0}^{M-1} a_k \geq (\prod_{k=0}^{M-1} a_k)^{1/M}.$$

En posant $a_k = \sigma^2_{Y_k} 2^{-2b_k}$, cette relation devient

$$\frac{1}{M}\sum_{k=0}^{M-1} \sigma^2_{Y_k}\ 2^{-2b_k} \geq (\prod_{k=0}^{M-1} \sigma^2_{Y_k} 2^{-2b_k})^{1/M} = (\prod_{k=0}^{M-1} \sigma^2_{Y_k})^{1/M}\ 2^{-2\sum_{k=0}^{M-1} b_k/M}$$

$$\frac{1}{M}\sum_{k=0}^{M-1} \sigma^2_{Y_k} 2^{-2b_k} \geq \alpha^2\ 2^{-2b}.$$

La valeur optimale est atteinte lorsque tous les termes intervenant dans la somme sont égaux. Dans ce cas, l'inégalité devient une égalité. Quel que soit k, on a

$$\sigma^2_{Y_k}\ 2^{-2b_k} = \alpha^2\ 2^{-2b} \tag{3.6}$$

soit

$$2^{b_k} = 2^b \sqrt{\frac{\sigma^2_{Y_k}}{\alpha^2}}. \tag{3.7}$$

On obtient ainsi l'équation (3.5).

L'équation (3.6) a une interpétation intéressante. Elle veut dire que la quantification scalaire optimale de chaque variable aléatoire $Y_k(m)$ entraîne la même erreur quadratique moyenne. On aboutit à la conclusion importante suivante : tout se passe comme si on cherchait à rendre l'erreur de reconstruction la plus proche possible d'un bruit blanc "pleine bande".

La puissance de l'erreur de quantification a pour expression

$$\boxed{\sigma^2_Q = c(1)\ (\prod_{k=0}^{M-1} \sigma^2_{Y_k})^{1/M}\ 2^{-2b}}. \tag{3.8}$$

3.3.3 Algorithme pratique

La formule (3.5) n'est pas exploitable directement. En effet, le nombre de bits b_k obtenu n'est pas forcément un nombre entier ; il n'est même pas assuré d'être un nombre positif. Une solution consiste à utiliser un algorithme sous-optimal[2] répartissant progressivement les bits là où ils ont le plus d'effet [3]

- Initialisation
  - $b_0 = \cdots = b_{M-1} = 0$
  - $\sigma^2_{Q_0} = \sigma^2_{Y_0} \cdots \sigma^2_{Q_{M-1}} = \sigma^2_{Y_{M-1}}$
- Tant que $\sum_{k=0}^{M-1} b_k < bM$
  - $l = \arg\max_k \sigma^2_{Q_k}$
  - $b_l = b_l + 1$
  - $\sigma^2_{Q_l} = \sigma^2_{Q_l}/4$

[2]C'est un algorithme de type "glouton" (greedy), adjectif utilisé chaque fois que le choix optimal est réalisé à chaque étape de l'algorithme dans l'espoir d'obtenir un résultat optimal global.

3.3.4 Compléments

Dans la présentation standard de l'allocation optimale de bits faite précédemment, on a écrit explicitement $\sigma^2_{Q_k}$ en fonction de $\sigma^2_{Y_k}$ et du nombre de bits b_k consacré à la quantification du signal de sous-bande $Y_k(m)$. La formule (3.3) réclame que l'on soit dans le cadre de l'hypothèse de haute résolution. Cette hypothèse est trop contraignante dans la pratique.

Supposons que pour chaque signal de sous-bande, on dispose d'un certain nombre de quantificateurs possibles et que l'on note $Q_k(i_k)$ le quantificateur sélectionné. On choisit, par exemple, de quantifier chaque signal $Y_k(m)$ par un quantificateur scalaire uniforme pour différentes résolutions. Dans ce cas très simple, le numéro i_k est la résolution b_k. On suppose, également, que pour chaque quantificateur on puisse en déduire (ou mesurer) la puissance de l'erreur de quantification $\sigma^2_{Q_k}(i_k)$ qu'il engendre. On note $b_k(i_k)$ le nombre de bits qu'il réclame. On verra au chapitre 4 qu'il est possible de réaliser un codage entropique après une quantification uniforme. Dans ce cas, on montrera que le nombre de bits nécessaire pour quantifier ce signal peut être diminué, ce qui expliquera la notation $b_k(i_k)$ et le fait que $b_k(i_k)$ puisse être non-entier.

Si l'on s'impose une combinaison quelconque de M quantificateurs $\underline{i} = [i_0, \cdots, i_{M-1}]^t$ parmi toutes les combinaisons possibles, on peut alors calculer la puissance de l'erreur de quantification

$$\sigma^2_Q(\underline{i}) = \frac{1}{M} \sum_{k=0}^{M-1} \sigma^2_{Q_k}(i_k) \tag{3.9}$$

et le nombre de bits moyen nécessaire pour quantifier les M signaux

$$b(\underline{i}) = \frac{1}{M} \sum_{k=0}^{M-1} b_k(i_k). \tag{3.10}$$

Ces deux résultats sont symbolisés par un point dans le plan σ^2_Q en fonction de b. L'ensemble de ces points forme un nuage qui n'est pas quelconque comme le montre la figure 3.2. La théorie débit-distorsion présentée (très succinctement) au chapitre 4 montrera qu'il existe vraisemblablement une enveloppe convexe. On recherche le point du nuage, ou de l'enveloppe convexe, minimisant $\sigma^2_Q(\underline{i})$ sous la contrainte que $b(\underline{i})$ soit inférieur ou égal à une valeur désirée b_d.

On procède en deux temps [12]. On recherche d'abord un point sur l'enveloppe convexe en fonction d'un paramètre λ puis on détermine la valeur optimale de ce paramètre.

Examinons la première optimisation. Soit la droite d'équation $\sigma^2_Q = -\lambda b + \alpha$. Pour une pente $-\lambda$ donnée, on cherche la droite, c'est à dire le paramètre α, passant par un point du nuage et présentant une intersection avec l'axe σ^2_Q la plus basse possible. Comme $\alpha(\underline{i}) = \sigma^2_Q(\underline{i}) + \lambda b(\underline{i})$ le point du nuage recherché est caractérisé par

$$\underline{i}_{opt}(\lambda) = \arg \min_{\underline{i}} [\sigma^2_Q(\underline{i}) + \lambda b(\underline{i})].$$

En utilisant les relations (3.9) et (3.10), on obtient

$$\underline{i}_{opt}(\lambda) = \arg \min_{\underline{i}} [\frac{1}{M} \sum_{k=0}^{M-1} \sigma^2_{Q_k}(i_k) + \lambda \frac{1}{M} \sum_{k=0}^{M-1} b_k(i_k)]$$

soit

$$\underline{i}_{opt}(\lambda) = \arg \sum_{k=0}^{M-1} \min_{i_k} [\sigma^2_{Q_k}(i_k) + \lambda b_k(i_k)].$$

On constate qu'il suffit de réaliser M minimisations indépendantes ce qui évite d'avoir à examiner toutes les combinaisons des différents quantificateurs. Prenons le cas où on serait amené à

sélectionner un quantificateur parmi 15 dans chacune des 32 sous-bandes comme on le verra par la suite dans le cas du codeur audio MPEG-1. La remarque précédente n'est pas anodine puisque l'on passe de 15^{32} comparaisons à 15×32 !

On procède maintenant à la deuxième optimisation. Toutes les droites déterminées par la minimisation précédente sont caractérisées par une équation qui ne dépend plus que de λ

$$\alpha[\underline{i}_{opt}(\lambda)] = \sigma_Q^2 + \lambda b. \tag{3.11}$$

Parmi toutes les droites possibles, il suffit de choisir celle qui maximise l'ordonnée du point d'intersection avec la droite $b = b_d$. Comme cette ordonnée β a pour expression

$$\beta(\lambda) = \alpha[\underline{i}_{opt}(\lambda)] - \lambda b_d$$

c'est à dire

$$\beta(\lambda) = \sigma_Q^2[\underline{i}_{opt}(\lambda)] + \lambda b[\underline{i}_{opt}(\lambda)] - \lambda b_d$$

il suffit de déterminer la valeur de λ qui maximise

$$\lambda_{opt} = \arg\max_{\lambda}[\sigma_Q^2[\underline{i}_{opt}(\lambda)] + \lambda b[\underline{i}_{opt}(\lambda)] - \lambda b_d].$$

Les M composantes de $i_{opt}(\lambda_{opt})$ caractérisent les M quantificateurs optimaux.

Considérons l'exemple suivant. On suppose qu'une transformée orthogonale décompose le signal en $M = 4$ sous-bandes. On se donne la puissance $\sigma_{Y_k}^2$ des signaux de sous-bandes par simple tirage d'une variable aléatoire. On imagine que l'on dispose de 4 quantificateurs possibles par sous-bande. Les quantificateurs sont des quantificateurs uniformes ; ils sont suivis par un codage entropique. On détermine les puissances $\sigma_{Q_k}^2(i_k)$ en faisant comme si la formule (3.3) était valide avec $b_k \in \{0, \cdots, 3\}$. Pour simuler le codage entropique et obtenir les expressions $b_k(i_k)$, on diminue légèrement de façon aléatoire la valeur b_k. La figure 3.2 visualise l'ensemble des points dont les coordonnées sont calculées à partir des formules (3.9) et (3.10) ainsi que les droites d'équation (3.11). Il s'agit de trouver quels sont les 4 quantificateurs qui engendrent le point d'abscisse inférieure à $b_d = 4$ et d'ordonnée σ_Q^2 la plus petite possible. La figure 3.3 montre

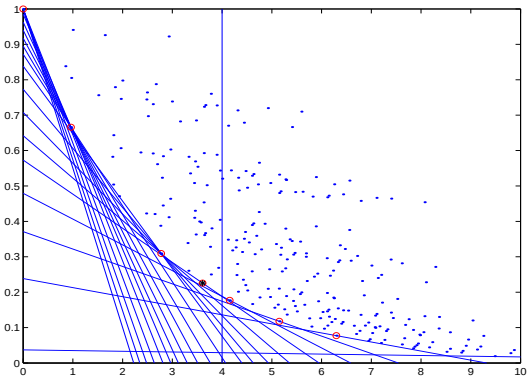

FIG. 3.2 – Ensemble des points de coordonnées $(\sigma_Q^2(\underline{i}), b(\underline{i}))$.

que la fonction $\beta(\lambda)$ est une fonction concave. Un simple algorithme du gradient permet donc d'obtenir λ_{opt}. On en déduit les M quantificateurs optimaux.

3.4 Transformation optimale

Dans un deuxième temps, on recherche parmi toutes les transformations $\mathcal{T}$ celle qui minimise σ_Q^2 après allocation optimale des bM bits disponibles sur le vecteur transformé. Pour minimiser

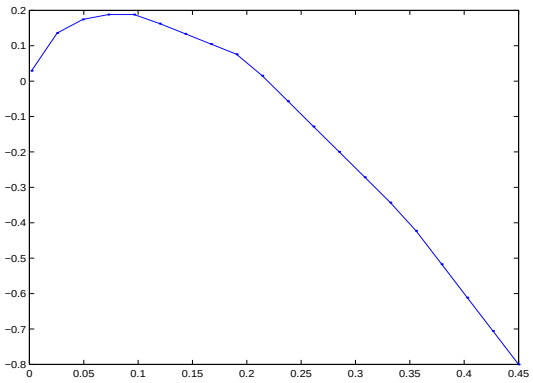

FIG. 3.3 – La fonction $\beta(\lambda)$ est une fonction concave.

σ_Q^2 donné par (3.8), il suffit donc de chercher la transformation $\mathcal{T}_{opt}$ qui minimise la moyenne géométrique des puissances des signaux de sous-bande. On se limite au cas des transformations orthogonales qui imposent déjà $N = M$.

Considérons la matrice de covariance du vecteur $\underline{X}(m)$, matrice de Toeplitz de dimension $M \times M$,

$$\mathbb{R}_X = \sigma_X^2 \begin{bmatrix} 1 & \rho_1 & \cdots & \rho_{M-1} \\ \rho_1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \rho_1 \\ \rho_{M-1} & \cdots & \rho_1 & 1 \end{bmatrix}.$$

Ecrivons la matrice de covariance du vecteur transformé $\underline{Y}(m)$ sous la forme

$$\mathbb{R}_Y = \begin{bmatrix} \sigma_{Y_0}^2 & \rho_{0,1}\sigma_{Y_0}\sigma_{Y_1} & \cdots & \rho_{0,M-1}\sigma_{Y_0}\sigma_{Y_{M-1}} \\ \rho_{1,0}\sigma_{Y_1}\sigma_{Y_0} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \rho_{M-2,M-1}\sigma_{Y_{M-2}}\sigma_{Y_{M-1}} \\ \rho_{M-1,0}\sigma_{Y_{M-1}}\sigma_{Y_0} & \cdots & \rho_{M-1,M-2}\sigma_{Y_{M-1}}\sigma_{Y_{M-2}} & \sigma_{Y_{M-1}}^2 \end{bmatrix}$$

en posant

$$\rho_{i,j} = \frac{\mathbb{E}\{Y_i Y_j\}}{\sqrt{\mathbb{E}\{Y_i^2\}\mathbb{E}\{Y_j^2\}}}.$$

On obtient

$$\mathbb{R}_Y = \begin{bmatrix} \sigma_{Y_0} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \sigma_{Y_{M-1}} \end{bmatrix} \mathbb{R}_Y' \begin{bmatrix} \sigma_{Y_0} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \sigma_{Y_{M-1}} \end{bmatrix}$$

avec

$$\mathbb{R}_Y' = \begin{bmatrix} 1 & \rho_{0,1} & \cdots & \rho_{0,M-1} \\ \rho_{1,0} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \rho_{M-2,M-1} \\ \rho_{M-1,0} & \cdots & \rho_{M-1,M-2} & 1 \end{bmatrix}.$$

Son déterminant a pour expression

$$\det \mathbb{R}_Y = \prod_{k=0}^{M-1} \sigma_{Y_k}^2 \ \det \mathbb{R}_Y'. \tag{3.12}$$

Comme

$$\det \mathbb{R}_Y = \det \mathbb{R}_X \det \mathcal{T}\mathcal{T}^t = \det \mathbb{R}_X$$

puisque la transformée est supposée orthogonale, on cherche la transformation $\mathcal{T}$ minimisant la moyenne géométrique des puissances, tout en maintenant constant le déterminant de $\mathbb{R}_Y$. La relation (3.12) montre que la transformation optimale est celle qui maximise le déterminant de la matrice $\mathbb{R}_Y^{'}$. Si l'on appelle $\lambda_0^{'} \cdots \lambda_{M-1}^{'}$ les valeurs propres de $\mathbb{R}_Y^{'}$, on sait [13] que ces valeurs propres sont réelles non négatives puisque $\mathbb{R}_Y^{'}$ est définie non négative et que la somme des valeurs propres est égale à M puisque la trace d'une matrice est invariante par transformation orthogonale. On a

$$\det \mathbb{R}_Y^{'} = \prod_{k=0}^{M-1} \lambda_k^{'} \leq (\frac{1}{M} \sum_{k=0}^{M-1} \lambda_k^{'})^M = 1 \tag{3.13}$$

en appliquant, une nouvelle fois, la propriété que la moyenne arithmétique d'un ensemble de réels non-négatifs est supérieure ou égale à la moyenne géométrique. L'égalité est atteinte lorsque toutes les valeurs propres sont égales à 1, c'est à dire lorsque les composantes du vecteur transformé sont décorrélées.

Si l'on appelle V la matrice carrée construite à partir des vecteurs propres de $\mathbb{R}_X$ et Λ la matrice diagonale construite à partir des valeurs propres, on obtient

$$\mathbb{R}_X V = V\Lambda.$$

On sait que la matrice V est une matrice orthogonale. On peut donc écrire

$$V^t \mathbb{E}\{\underline{X}\underline{X}^t\} V = \Lambda$$

$$\mathbb{E}\{V^t \underline{X} (V^t \underline{X})^t\} = \Lambda.$$

Le vecteur transformé par la transformation $\mathcal{T} = V^t$ a ses composantes décorrélées. C'est donc la transformation optimale. On appelle cette transformation la transformation de Karhunen-Loeve.

La transformée de Karhunen-Loeve réalise donc une décomposition du vecteur $\underline{X}(m)$ sur les vecteurs propres de la matrice de covariance. Elle produit des coefficients décorrélés et on peut montrer que c'est la transformée qui réussit à concentrer le maximum de puissance dans le plus petit nombre de coefficients. Toutefois elle a deux inconvénients très importants. La charge de calcul est souvent prohibitive parce que cette transformée dépend du signal et que l'algorithme de calcul des valeurs propres et des vecteurs propres d'une matrice, basé sur des factorisations QR successives, est très lourd. Cette transformée présente également l'inconvénient de ne pas avoir d'interprétation fréquentielle simple.

3.5 Performances

3.5.1 Gain de transformation

Le fait de réaliser une transformation quelconque $\underline{y}(m) = \mathcal{T}\underline{x}(m)$ puis une allocation de bits optimale apporte déjà un gain puisque

$$G_t = \frac{c(1)\sigma_X^2 2^{-2b}}{c(1)(\prod_{k=0}^{M-1} \sigma_{Y_k}^2)^{1/M} 2^{-2b}} = \frac{\frac{1}{M}\sum_{k=0}^{M-1} \sigma_{Y_k}^2}{(\prod_{k=0}^{M-1} \sigma_{Y_k}^2)^{1/M}}$$

apparaît comme le rapport d'une moyenne arithmétique sur une moyenne géométrique de nombres positifs et ne peut être que supérieur ou égal à 1. Ce gain[3] dû à la transformation et à l'allocation de bits optimale est fonction de $\mathcal{T}$ et de M.

[3]Dans la littérature des codeurs d'image, on emploie habituellement le terme de gain de codage qui n'a bien sûr rien à voir avec le gain de codage employé en codage de canal.

Si, en plus, on applique la transformée de Karhunen-Loeve, sachant que on a toujours

$$\sigma_X^2 = \frac{1}{M}\sum_{k=0}^{M-1}\lambda_k$$

où $\lambda_0 \cdots \lambda_{M-1}$ sont les valeurs propres de $\mathbb{R}_X$ relativement à des vecteurs propres normalisés mais que

$$(\prod_{k=0}^{M-1}\sigma_{Y_k}^2)^{1/M} = (\prod_{k=0}^{M-1}\lambda_k)^{1/M}$$

n'a lieu que dans ce cas, on remarque que le gain de transformation est égal au rapport de la moyenne arithmétique et de la moyenne géométrique des valeurs propres de la matrice de covariance $\mathbb{R}_X$

$$G_t(M) = \frac{\frac{1}{M}\sum_{k=0}^{M-1}\lambda_k}{(\prod_{k=0}^{M-1}\lambda_k)^{1/M}}.$$

La transformée de Karhunen-Loeve est la transformation qui entraîne le gain de transformation le plus élevé.

La moyenne géométrique s'écrit

$$(\prod_{k=0}^{M-1}\lambda_k)^{1/M} = \exp(\frac{1}{M}\sum_{k=0}^{M-1}\log_e\lambda_k).$$

Le gain de transformation a donc pour expression

$$G_t(M) = \frac{\frac{1}{M}\sum_{k=0}^{M-1}\lambda_k}{\exp(\frac{1}{M}\sum_{k=0}^{M-1}\log_e\lambda_k)}.$$

On montre deux résultats [14].

- Si λ_k est une valeur propre de la matrice de covariance $\mathbb{R}_X$ du processus $X(n)$ et si $S_X(f)$ est la densité spectrale de ce processus, alors

  $$\lambda_k = S_X(\Omega_k).$$

  Les M fréquences $\Omega_0 \cdots \Omega_{M-1}$ sont appelées les fréquences propres. Elles ne possèdent pas la propriété d'être uniformément réparties dans l'intervalle $[0, 1]$.
- Quelle que soit une fonction $g(.)$, on a

  $$\lim_{M\to\infty}\frac{1}{M}\sum_{k=0}^{M-1}g(\lambda_k) = \int_{-1/2}^{1/2}g[S_X(f)]df.$$

  Ce résultat est connu sous le nom de théorème de Szego.

Connaissant ces deux résultats, on en déduit que

$$G_t(\infty) = \frac{\int_{-1/2}^{1/2}S_X(f)df}{\exp(\int_{-1/2}^{1/2}\log_e[S_X(f)]df)}.$$

On reconnaît l'expression du gain de prédiction donnée par la formule (1.13). On a donc

$$G_t(\infty) = G_p(\infty).$$

Asymptotiquement les techniques de quantification basées sur le codage par transformée et sur la quantification prédictive, donnent des résultats identiques. Toutefois, pour une valeur donnée

du paramètre M, la quantification prédictive à l'ordre M peut paraître préférable à l'utilisation de la transformée optimale de dimension M car le quantificateur par transformée exploite la corrélation contenue dans le signal jusqu'à l'ordre M simplement. Par contre, dans le quantificateur prédictif, il y a exploitation de la corrélation à un ordre supérieur grâce à la boucle de prédiction. Si le processus à quantifier est autorégressif d'ordre P, il suffit de réaliser une prédiction à l'ordre P pour que le gain de prédiction atteigne sa valeur asymptotique. Ce n'est pas le cas du quantificateur par transformée.

On peut remarquer qu'employer la transformée de Karhunen-Loeve consiste à chercher à faire en sorte que $\{\sigma^2_{Y_0} \cdots \sigma^2_{Y_{M-1}}\}$ soit une estimation spectrale la plus fidèle possible de $S_X(f)$.

3.5.2 Résultats de simulation

On quantifie le signal représenté figure 2.1 avec une résolution de 2 bits par échantillon. On montre figure 3.4 les partitions et les représentants correspondant à un quantificateur scalaire non-uniforme, un quantificateur vectoriel et un quantificateur scalaire avec allocation optimale de bits précédée d'une transformation de Karhunen-Loeve (pour $N = M = 2$, il s'agit d'une simple rotation des axes de 45 degrés).

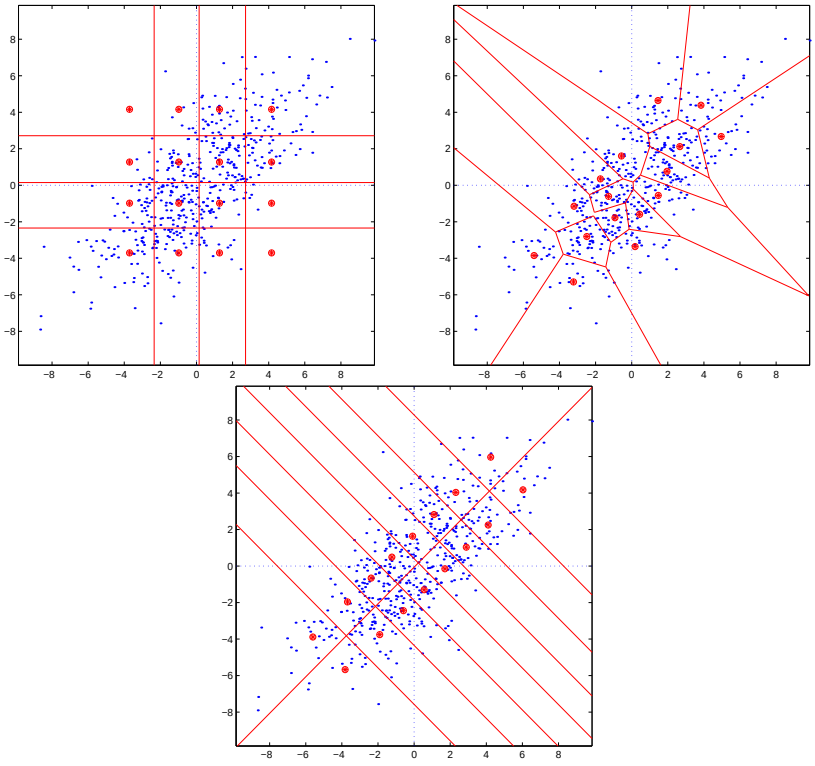

FIG. 3.4 – Différentes méthodes de quantification lorsque $b = 2$.

Chapitre 4

Codage entropique

4.1 Introduction

Considérons un signal $x(t)$ à temps continu et à bande limitée $[-B, +B]$. On l'échantillonne à une fréquence supérieure ou égale à la fréquence de Nyquist $f_e = 2B$. On obtient, sans perte d'information, un signal à temps discret $x(n)$. On interprétera ce signal comme la réalisation d'un processus aléatoire à temps discret $X(n)$. On suppose que ce processus aléatoire possède les bonnes propriétés habituelles de stationnarité et d'ergodicité.

Ce processus est à valeurs continues. Supposons qu'il ait été ensuite quantifié avec une résolution importante. Ce processus aléatoire devient un processus aléatoire à valeurs discrètes, c'est-à-dire que $X(n)$ prend ses valeurs dans un ensemble fini. En théorie de l'information, on appelle $X(n)$ la source d'information, l'ensemble fini comprenant L_X éléments l'alphabet d'entrée $A_X = \{x^1 \cdots x^{L_X}\}$ et les éléments x^i les symboles d'entrée ou lettres de l'alphabet. On cherche maintenant à comprimer cette information. L'objet de ce chapitre est d'expliquer d'abord que, moyennant certaines hypothèses, il est possible de réaliser cette opération sans apporter de distorsion. On parle alors de codage sans bruit, sans perte ou de codage entropique. Malheureusement les taux de compression permis sont généralement insuffisants. On acceptera donc une certaine distorsion. On montrera qu'il existe une fonction appelée fonction débit-distorsion donnant une limite inférieure pour la distorsion lorsque l'on s'impose le débit ou inversement une limite inférieure pour le débit lorsque l'on s'impose la distorsion. On introduira également dans ce chapitre la notion de capacité d'un canal de transmission de façon à pouvoir énoncer le théorème du codage combiné source-canal. On réexaminera la procédure de quantification scalaire en introduisant un nouveau mode de quantification, le quantificateur scalaire avec contrainte entropique, que l'on comparera au quantificateur examiné dans les chapitres précédents, le quantificateur de Lloyd-Max.

Le développement présenté dans ce chapitre est plutôt sommaire. La plupart des théorèmes ne sont pas démontrés. On laissera le lecteur consulter les travaux de C. Shannon [15, 16] sur lesquels est basée la théorie de l'information et les ouvrages de référence [17, 18, 19, 5], plus particulièrement [20]. Ce développement est également incomplet. Par exemple, il ne présente pas l'algorithme de Ziv-Lempel [20] largement utilisé pour comprimer des fichiers informatiques grâce à la commande UNIX *compress*.

4.2 Codage sans bruit d'une source discrète sans mémoire

On suppose, dans une première étape, que le processus $X(n)$ est une suite de variables aléatoires indépendantes et identiquement distribuées (suite i.i.d.) prenant ses valeurs dans $A_X =$

$\{x^1 \cdots x^{L_X}\}$. On parle alors de source discrète sans mémoire. On note

$$p_X(i) = P_{rob}\{X(n) = x^i\}$$

la distribution des différentes probabilités. Cette distribution est indépendante de l'instant d'observation n puisque le processus aléatoire est stationnaire.

4.2.1 Entropie d'une source

On appelle information propre de l'événement $\{X(n) = x^i\}$ la quantité

$$I(x^i) = -\log_2 p_X(i).$$

Elle est positive ou nulle. Elle est exprimée en bits/symbole[1] puisque l'on a pris un logarithme en base 2. On appelle information propre moyenne ou entropie de la source $X(n)$ l'espérance de la variable aléatoire $I(x^i)$

$$H(X) = E\{I(x^i)\}$$

$$\boxed{H(X) = -\sum_{i=1}^{L_X} p_X(i) \log_2 p_X(i)}. \tag{4.1}$$

L'entropie[2] est la quantité d'information qu'apporte, en moyenne, une réalisation de $X(n)$. Elle donne le nombre de bits nécessaires en moyenne pour décrire complètement la source. Elle mesure l'incertitude associée à la source.

Prenons l'exemple d'une source binaire, $X(n) \in \{x^1, x^2\}$, sans mémoire et notons pour simplifier les notations $p = p_X(1)$. L'entropie de cette source vaut

$$H(X) = -p \log_2 p - (1-p) \log_2 (1-p).$$

Elle est visualisée figure 4.1. L'entropie est positive ou nulle. Elle est nulle si $p = 0$ ou $p = 1$, c'est-à-dire si la source est totalement prédictible. Elle est maximale et égale à 1 si les deux symboles sont équiprobables. C'est une fonction concave de p.

La démarche proposée par Shannon dans son papier fondateur [15] pour justifier cette définition peut être très schématiquement résumée de la façon suivante. Considérons la source binaire, $X(n) \in \{x^1, x^2\}$, supposée sans mémoire observée sur N instants. Un tirage aléatoire fournit une réalisation particulière, par exemple $[x^2 x^1 x^1 x^2 \cdots x^2]$. Si on compte le nombre de fois N_1 où apparaît x^1 et le nombre de fois N_2 où apparaît x^2, la loi (faible) des grands nombres nous dit que si N est assez grand alors $N_1/N \approx p$ et $N_2/N \approx 1-p$. Puisque $X(n)$ est une suite de variables aléatoires indépendantes, alors la probabilité d'obtenir la réalisation observée est égale à

$$Prob\{\underline{X} = [x^2 x^1 x^1 x^2 \cdots x^2]\} = p^{N_1}(1-p)^{N_2} = p^{Np}(1-p)^{N(1-p)}.$$

Comme cette probabilité est approximativement constante (cette propriété est appelée dans la littérature "propriété d'équirépartition asymptotique"), cela veut dire que le nombre de réalisations différentes pouvant être obtenues (on parle de "séquences typiques") est donnée par

$$N_{st} = \left(p^{Np}(1-p)^{N(1-p)}\right)^{-1}.$$

[1] Si l'on suivait les recommandations de l'UIT-T, on parlerait de shannon/symbole pour ne pas, justement, le confondre avec le bit qui est l'unité de comptage des symboles binaires employer pour coder une information. On utilise ici la terminologie usuelle.

[2] La notation habituelle $H(X)$ ne veut pas dire que H est une fonction des valeurs prises par la variable aléatoire X. L'entropie ne dépend que de la distribution des probabilités $p_X(i)$.

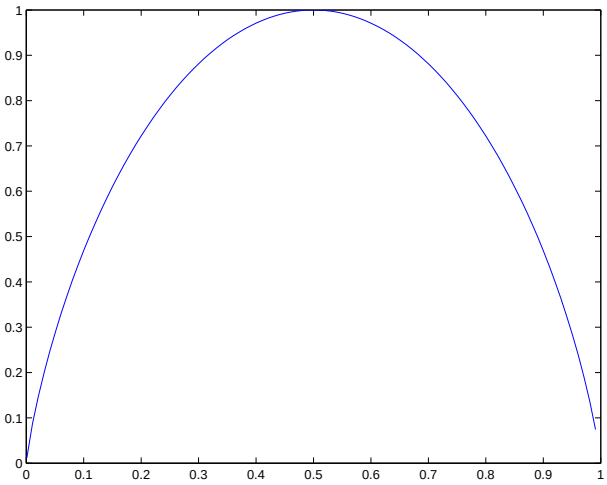

FIG. 4.1 – Entropie d'une source binaire sans mémoire en fonction de la probabilité p d'obtenir un des 2 symboles.

Comme

$$\log_2 p^{-Np}(1-p)^{-N(1-p)} = -Np\log_2 p - N(1-p)\log_2(1-p) = NH(X)$$

et que le nombre total de séquences est égal à L_X^N, on obtient

$$N_{st} = 2^{NH(X)} \leq L_X^N = 2^{N\log_2(L_X)}.$$

Ceci veut simplement dire que, parmi toutes les séquences possibles, un certain nombre est de probabilité nulle. Il est inutile de prévoir des bits pour coder les séquences qui ne seront jamais réalisées !

Tous les résultats présentés pour $L_X = 2$ se généralisent lorsque $L_X > 2$. L'expression (4.1) montre que l'entropie d'une source discrète est positive ou nulle. Elle est nulle lorsque $X(n)$ est presque sûrement égale à une valeur particulière x^i ; la source est alors totalement prédictible. L'entropie est maximale et vaut $\log_2 L_X$ lorsque les valeurs de l'alphabet sont équiprobables. On a

$$0 \leq H(X) \leq \log_2 L_X.$$

4.2.2 Codage d'une source

Définitions

Dans les chapitres précédents, on n'a pas cherché à définir de façon précise le mot codage. On a simplement dit que l'on cherchait à associer à un scalaire $x(n)$ ou à un vecteur $\underline{x}(m) = [x(mN)\cdots x(mN+N-1)]^t$, un nombre $i(n)$ ou $i(m) \in \{1\cdots L\}$ avec $L = 2^{bN}$, le paramètre b spécifiant le nombre de bits par échantillon. Dans ce chapitre, l'information à transmettre ou à stocker, modélisée par le processus aléatoire $X(n)$, prend ses valeurs dans un ensemble fini, l'alphabet d'entrée A_X, et on désire représenter (coder) les différents éléments de cet ensemble de façon adaptée aux caractéristiques du canal de transmission et de façon efficace.

De façon adaptée aux caractéristiques du canal de transmission veut dire que la représentation de chaque symbole d'entrée, un mot du code, peut être construite à partir d'éléments d'un autre alphabet adapté au canal. On supposera par la suite que cet alphabet est composé de deux éléments $A_C = \{a^1, a^2\}$, par exemple les deux symboles binaires habituels 0 et 1.

De façon efficace veut dire que l'on cherche à représenter la source en utilisant le minimum de bits, c'est-à-dire en minimisant la longueur moyenne des mots du code. Précédemment, tous les mots du code avaient la même longueur bN. Maintenant, on va très naturellement associer aux symboles d'entrée les plus probables, les mots de code les plus courts. Le code est dit à longueur variable.

Plus précisément, on appelle codage de la source $X(n)$ une application de l'alphabet A_X dans l'ensemble des suites finies d'éléments de l'alphabet A_C. Le code $C = \{c^1 \cdots c^{L_X}\}$ est l'ensemble de ces suites. Chaque suite possible $c^i = [a^{i(1)} \cdots a^{i(l)}]$ est un mot du code. Le nombre d'éléments de A_C composant un mot est la longueur l du mot. La longueur moyenne des mots du code est donnée par

$$\bar{l} = \sum_{i=1}^{L_X} p_X(i) l(c^i).$$

Code instantané uniquement décodable

Remarquons tout d'abord qu'il semble exister un problème associé à un code de longueur variable : faut-il ajouter des séparateurs entre mots du code dans une séquence ? Cela n'est pas nécessaire si le code vérifie la condition dite du préfixe : aucun mot du code ne doit être un préfixe d'un autre mot du code.

Prenons l'exemple d'une source ne pouvant prendre que six valeurs différentes $\{x^1, \cdots, x^6\}$. Le code défini par les associations données table 4.1 est un code valide, on dit uniquement

Symboles	x^1	x^2	x^3	x^4	x^5	x^6
Codes	0	100	101	110	1110	1111

TAB. 4.1 – Définition d'un code.

décodable, puisque, recevant par exemple la chaîne binaire 11001110101110, on en déduit la séquence x^4, x^1, x^5, x^3, x^4. On constate que le décodage des symboles se fait sans référence aux mots de code futurs. Le code est dit instantané. On se limitera par la suite à ce cas particulier.

Le code précédent vérifie aussi la condition du préfixe. Une façon simple de vérifier la condition du préfixe ou de construire un code vérifiant cette condition est de tracer un graphe orienté en forme d'arbre binaire, d'étiqueter chaque branche partant d'un noeud par les symboles 0 ou 1 et d'associer un mot du code à chaque noeud terminal en prenant comme mot de code la succession des symboles binaires sur les branches comme le montre la figure 4.2. Cet arbre n'est pas forcément complet. Le nombre de branches entre le noeud initial, la racine, et un noeud terminal, une feuille, spécifie la longueur du code associé.

Inégalité de Kraft

Considérons un code binaire $C = \{c^1 \cdots c^{L_X}\}$ représentant sans erreur une source. Une condition nécessaire et suffisante pour que ce code vérifie la condition du préfixe est que

$$\sum_{i=1}^{L_X} 2^{-l(c^i)} \leq 1 \tag{4.2}$$

où $l(c^i)$ est la longueur du mot de code c^i. Par exemple, sur la figure 4.2, on trouve

$$2^{-1} + 3 \times 2^{-3} + 2 \times 2^{-4} = 1.$$

Donnons simplement le principe de la démonstration de la condition nécessaire lorsque L_X est une puissance de 2. Considérons l'arbre complet représentant l'ensemble des L_X mots d'un

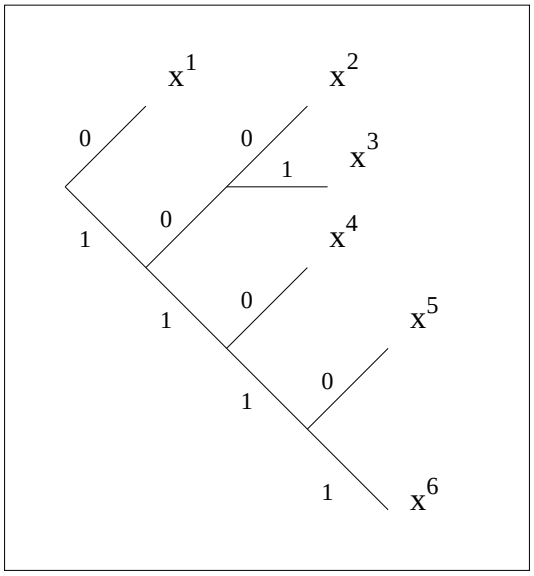

FIG. 4.2 – Arbre binaire associé à un code uniquement décodable.

code C' dont tous les mots du code auraient la même longueur l_{max} telle que

$$L_X = 2^{l_{max}}.$$

Ce code vérifie la relation (4.2) puisque

$$\sum_{i=1}^{L_X} 2^{-l_{max}} = 2^{l_{max}} 2^{-l_{max}} = 1.$$

On passe de l'arbre associé au code C' à l'arbre associé au code C en élaguant un certain nombre de branches et en créant de nouvelles. Elaguer des branches ne modifie pas le premier membre de (4.2) puisque l'on sera toujours amené à remplacer deux termes de la forme 2^{-l} par 2^{-l+1}. Il en sera de même si l'on crée de nouvelles branches puisque l'on remplacera 2^{-l} par $2 \times 2^{-l-1}$.

Code optimal

Relâchons la contrainte suivant laquelle les quantités $l(c^i)$ doivent être des entiers et remplaçons le signe d'inégalité par une égalité dans (4.2). Le code optimal est celui qui minimise la longueur moyenne

$$\bar{l} = \sum_{i=1}^{L_X} p_X(i) l(c^i)$$

sous la contrainte

$$\sum_{i=1}^{L_X} 2^{-l(c^i)} = 1.$$

Introduisons un multiplieur de Lagrange et écrivons pour $i = 1 \cdots L_X$

$$\frac{\partial}{\partial l(c^i)} [\sum_{j=1}^{L_X} p_X(j) l(c^j) + \lambda \sum_{j=1}^{L_X} 2^{-l(c^j)}] = 0.$$

On obtient

$$p_X(i) - \lambda \log_e 2 \; 2^{-l(c^i)} = 0$$

soit

$$2^{-l(c^i)} = \frac{p_X(i)}{\lambda \log_e 2}.$$

Comme

$$\sum_{i=1}^{L_X} 2^{-l(c^i)} = \frac{1}{\lambda \log_e 2} \sum_{i=1}^{L_X} p_X(i) = 1$$

cela impose la valeur de la constante $\lambda \log_e 2 = 1$. On obtient

$$2^{-l(c^i)} = p_X(i).$$

$$l(c^i) = -\log_2 p_X(i).$$

La longueur moyenne correspondant au code optimal est donnée par

$$\bar{l} = \sum_{i=1}^{L_X} p_X(i) l(c^i) = -\sum_{i=1}^{L_X} p_X(i) \log_2 p_X(i)$$

$$\bar{l} = H(X).$$

Parmi tous les codes vérifiant la condition du préfixe, celui qui minimise la longueur moyenne des mots du code a une longueur moyenne égale à l'entropie de la source. L'entropie $H(X)$ apparaît donc comme une limite fondamentale pour représenter sans distorsion une source d'information.

4.2.3 Théorème du codage sans bruit d'une source discrète sans mémoire

Le développement qui suit a simplement pour but de préciser le résultat précédent lorsque l'on impose aux longueurs des mots du code d'être des valeurs entières. Ces résultats ont été démontrés par Shannon en 1948 [15].

Proposition 1

Si $p(1) \cdots p(L)$ et $q(1) \cdots q(L)$ sont deux distributions de probabilité quelconques, alors

$$\sum_{i=1}^{L} p(i) \log_2 \frac{q(i)}{p(i)} \leq 0.$$

L'égalité a lieu si et seulement si $p(i) = q(i)$ quel que soit $i = 1 \cdots L$.

En effet, on sait que, pour tout x positif,

$$\log_2 x \leq (x-1) \log_2 e.$$

Donc

$$\log_2 \frac{q(i)}{p(i)} \leq (\frac{q(i)}{p(i)} - 1) \log_2 e$$

ou

$$\sum_{i=1}^{L} p(i) \log_2 \frac{q(i)}{p(i)} \leq \sum_{i=1}^{L} p(i) (\frac{q(i)}{p(i)} - 1) \log_2 e = 0.$$

L'égalité est atteinte si et seulement si $q(i)/p(i) = 1$ quel que soit i.

L'expression

$$D(p||q) = \sum_{i=1}^{L} p(i) \log_2 \frac{p(i)}{q(i)} \tag{4.3}$$

s'appelle l'entropie relative ou distance de Kullback-Leibler entre deux distributions de probabilité. Elle est toujours positive ou nulle. Elle s'interprète comme une mesure de distance entre deux distributions de probabilité bien que cela ne soit pas, à proprement parler, une distance puisque ce n'est pas une expression symétrique et qu'elle ne respecte pas l'inégalité triangulaire !

Proposition 2

Tout codage de la source $X(n)$ par un code instantané uniquement décodable entraîne une longueur moyenne vérifiant

$$H(X) \leq \bar{l}.$$

En effet, tout code instantané uniquement décodable vérifie l'inégalité de Kraft

$$\sum_{i=1}^{L_X} 2^{-l(c^i)} \leq 1.$$

On crée une nouvelle distribution de probabilité de la forme

$$q(i) = a 2^{-l(c^i)}$$

avec $a \geq 1$ puisque

$$\sum_{i=1}^{L_X} q(i) = 1 = a \sum_{i=1}^{L_X} 2^{-l(c^i)}.$$

La proposition 1 dit que

$$\sum_{i=1}^{L_X} p_X(i) \log_2 a \frac{2^{-l(c^i)}}{p_X(i)} \leq 0$$

soit

$$-\sum_{i=1}^{L_X} p_X(i) \log_2 p_X(i) - \sum_{i=1}^{L_X} p_X(i) l(c^i) + \log_2 a \leq 0$$

$$H(X) \leq \bar{l} - \log_2 a.$$

On obtient donc la formule désirée puisque $\log_2 a \geq 0$.

Proposition 3

Il existe un code instantané uniquement décodable vérifiant

$$\bar{l} < H(X) + 1.$$

En effet, choisissons un code tel que

$$-\log_2 p_X(i) \leq l(c^i) < -\log_2 p_X(i) + 1.$$

A partir de la première inégalité

$$-l(c^i) \leq \log_2 p_X(i)$$

on obtient

$$2^{-l(c^i)} \leq p_X(i)$$

$$\sum_{i=1}^{L_X} 2^{-l(c^i)} \leq \sum_{i=1}^{L_X} p_X(i) = 1.$$

On en déduit l'existence d'un code ayant cette distribution des longueurs. A partir de la deuxième inégalité, on obtient

$$\sum_{i=1}^{L_X} p_X(i) l(c^i) < -\sum_{i=1}^{L_X} p_X(i) \log_2 p_X(i) + 1$$

$$\bar{l} < H(X) + 1.$$

Théorème

Pour toute source discrète sans mémoire $X(n)$, il existe un code instantané représentant exactement cette source et uniquement décodable vérifiant

$$H(X) \leq \bar{l} < H(X) + 1 \tag{4.4}$$

où $H(X)$ est l'entropie de la source et $\bar{l}$ la longueur moyenne du code.

4.2.4 Construction d'un code

Code de Shannon

La façon la plus simple de procéder est de choisir

$$l(c^i) = \lceil -\log_2 p_X(i) \rceil$$

où $\lceil x \rceil$ représente le plus petit entier supérieur ou égal à x. On a

$$\bar{l} = \sum_{i=1}^{L_X} p_X(i) \lceil -\log_2 p_X(i) \rceil$$

$$\bar{l} \leq -\sum_{i=1}^{L_X} p_X(i) \log_2 p_X(i) + \sum_{i=1}^{L_X} p_X(i)$$

$$\bar{l} \leq H(X) + 1.$$

Comme

$$2^{-\lceil -\log_2 p_X(i) \rceil} \leq 2^{\log_2 p_X(i)}$$

$$\sum_{i=1}^{L_X} 2^{-l(c^i)} \leq \sum_{i=1}^{L_X} p_X(i)$$

$$\sum_{i=1}^{L_X} 2^{-l(c^i)} \leq 1$$

on en déduit qu'il existe un code instantané vérifiant la condition du préfixe.

Algorithme de Huffman

L'algorithme qu'a proposé Huffman en 1951 consiste à construire progressivement un arbre binaire en partant des noeuds terminaux.

- On part des deux listes $\{x^1 \cdots x^{L_X}\}$ et $\{p_X(1) \cdots p_X(L_X)\}$.
- On sélectionne les deux symboles les moins probables, on crée deux branches dans l'arbre et on les étiquette par les deux symboles binaires 0 et 1.
- On actualise les deux listes en rassemblant les deux symboles utilisés en un nouveau symbole et en lui associant comme probabilité la somme des deux probabilités sélectionnées.
- On recommence les deux étapes précédentes tant qu'il reste plus d'un symbole dans la liste.

On montre que cet algorithme est l'algorithme optimal. Pour aucun autre code uniquement décodable, la longueur moyenne des mots du code est inférieure.

Symboles	x^1	x^2	x^3	x^4	x^5	x^6
Probabilités	0,5	0,15	0,17	0,08	0,06	0,04

TAB. 4.2 – Probabilités associées aux six événements $\{X(n) = x^i\}$.

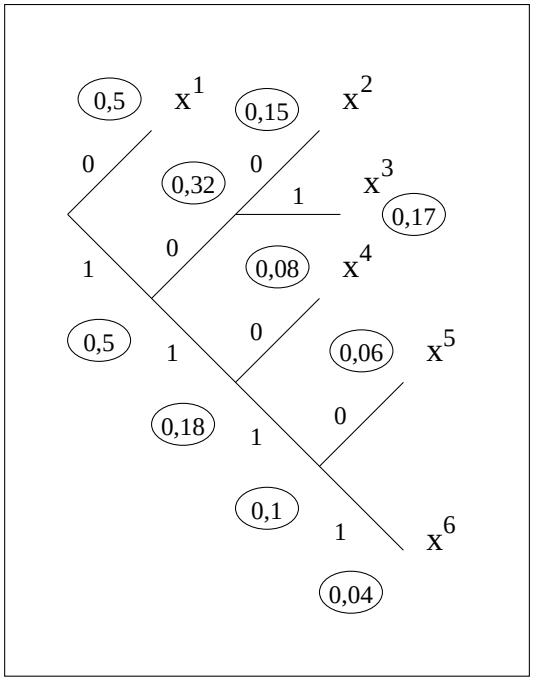

FIG. 4.3 – Illustration de l'algorithme de Huffman.

Premier exemple

Prenons l'exemple d'une source ne pouvant prendre que six valeurs différentes et supposons connues les probabilités. Elles sont données table 4.2. L'entropie de cette source est égale à 2,06 bits. Le code de Shannon entraîne une longueur moyenne égale à 2,28 bits. L'algorithme de Huffman fournit l'arbre binaire schématisé figure 4.3. La table de codage est indiquée table 4.1. La longueur moyenne est égale à 2,1 bits, valeur très voisine de la limite théorique.

4.2.5 Généralisation

La double inégalité (4.4) est trop imprécise car la valeur de $H(X)$ est généralement faible. Pour diminuer cette imprécision, on forme un vecteur aléatoire, noté $\underline{X}_N$, en regroupant N variables aléatoires $X(mN)\cdots X(mN+N-1)$ et on cherche à associer à toute réalisation possible de ce vecteur un mot du code. Cela correspond, dans ce cas, à une application de l'ensemble produit $A_X \times \cdots \times A_X$ dans l'ensemble des suites finies. On conserve, dans ce paragraphe, l'hypothèse que la source est sans mémoire.

Théorème

On montre que si on regroupe N symboles de la source et si on lui associe un mot du code c^i de longueur $l(c^i)$, alors il un existe un code tel que la longueur moyenne

$$\bar{l} = \sum_{\underline{X}_N} Pr(\underline{X}_N) l(c^i)$$

vérifie

$$H(X) \leq \frac{\bar{l}}{N} < H(X) + \frac{1}{N}. \tag{4.5}$$

Le rapport $\bar{l}/N$ représente le nombre moyen de bits par symbole nécessaire et suffisant pour pouvoir représenter exactement la source.

Il existe un autre théorème consistant à supposer que la longueur de tous les mots du code est identique et à permettre à N de varier. Ce théorème dit que quel que soit $\epsilon > 0$, il existe un code tel que si

$$\frac{l}{N} \geq H(X) + \epsilon$$

alors la probabilité p_e pour qu'à une séquence $X(mN) \cdots X(mN + N - 1)$ on ne puisse pas lui associer un mot du code, peut être rendue arbitrairement petite pourvu que N soit suffisamment grand. Ce théorème dit également que si

$$\frac{l}{N} \leq H(X) - \epsilon$$

il n'existe pas de code pour lequel la probabilité p_e puisse être rendue arbitrairement petite.

Deuxième exemple

Supposons une source sans mémoire ne pouvant prendre que deux valeurs de probabilités connues. Formons un vecteur de dimension 2 puis un vecteur de dimension 3. L'ensemble des probabilités des différents événements est donné table 4.3. L'entropie de cette source est égale

			x^1	x^2			
			0,7	0,3			
		x^1x^1	x^1x^2	x^2x^1	x^2x^2		
		0,49	0,21	0,21	0,09		
$x^1x^1x^1$	$x^1x^1x^2$	$x^1x^2x^1$	$x^1x^2x^2$	$x^2x^1x^1$	$x^2x^1x^2$	$x^2x^2x^1$	$x^2x^2x^2$
0,343	0,147	0,147	0,063	0,147	0,063	0,063	0,027

TAB. 4.3 – Probabilités associées aux 2, 4 et 8 événements possibles.

à 0,88 bit. Les nombres moyens de bits par symbole sont donnés table 4.4. Ils ne créent pas

N	1	2	3
$\bar{l}/N$	1	0,905	0,908

TAB. 4.4 – Nombres moyens de bits par symbole.

forcément une suite décroissante lorsque N augmente mais ils vérifient la double inégalité (4.5).

4.2.6 Codage arithmétique

Il n'est plus question de rechercher un code instantané comme précédemment. Cette technique de codage code simultanément plusieurs symboles à la fois.

Considérons cinq symboles dont on suppose connue la distribution des probabilités donnée table 4.5 et associons à chaque symbole une partie de l'intervalle $[0-1[$ en fonction de sa probabilité comme indiqué dans cette table. Coder (simultanément) par exemple les trois symboles $[x^3, x^2, x^4]$ consiste à rechercher un sous-ensemble de l'intervalle $[0-1[$ de la façon suivante.

- On sélectionne d'abord l'intervalle $[0,3-0,4[$ associé au symbole x^3.
- On "inclut" ensuite l'intervalle $[0,1-0,3[$ spécifique du symbole x^2 dans l'intervalle $[0,3-0,4[$ ce qui donne l'intervalle $[0,31-0,33[$.

Symboles	x^1	x^2	x^3	x^4	x^5
Probabilités	0,1	0,2	0,1	0,4	0,2
Intervalles	[0 - 0,1[	[0,1 - 0,3[	[0,3 - 0,4[	[0,4 - 0,8[	[0,8 - 1[

TAB. 4.5 – Probabilités associées aux cinq événements $\{X(n) = x^i\}$ et définition des intervalles de codage.

– On poursuit avec x^4 ce qui donne les deux bornes du nouvel intervalle

$$0,31 + 0,4 * (0,33 - 0,31) = 0,318$$

$$0,31 + 0,8 * (0,33 - 0,31) = 0,326.$$

Coder $[x^3, x^2, x^4]$ revient à transmettre n'importe quel réel appartenant à l'intervalle $[0,318 - 0,326[$ par exemple 0,32 ou plus précisément celui qui a la représentation binaire la plus économique possible par exemple $\{0101001\}$ puisque $2^{-2} + 2^{-4} + 2^{-7} = 0,3203$.

Au décodeur, on réalise le traitement suivant

– Puisque 0,3203 appartient à l'intervalle $[0,3 - 0,4[$ on décide x^3.

– Comme

$$\frac{0,3203 - 0,3}{0,4 - 0,3} = 0,202 \in [0,1 - 0,3[$$

on décide x^2.

– Etc.

Pour plus de détails, en particulier pour comprendre comment dans la pratique on obtient les représentations binaires les plus économiques, on consultera l'article [21].

4.3 Codage sans bruit d'une source discrète avec mémoire

On a examiné dans le développement précédent le cas le plus élémentaire. Généralement il existe une dépendance statistique entre les échantillons successifs de la source que l'on va chercher à exploiter pour réduire le nombre de bits nécessaire pour représenter exactement la source.

4.3.1 Nouvelles définitions

Considérons deux variables aléatoires discrètes[3] X et Y prenant leurs valeurs dans respectivement $A_X = \{x^1 \cdots x^{L_X}\}$ et $A_Y = \{y^1 \cdots y^{L_Y}\}$ et notons les probabilités conjointes

$$p_{XY}(i,j) = P_{rob}\{X = x^i, Y = y^j\}$$

et les probabilités conditionnelles

$$p_{Y|X}(i|j) = P_{rob}\{Y = y^j | X = x^i\}.$$

On appelle

– information conjointe des 2 événements $\{X = x^i\}$ et $\{Y = y^j\}$, la quantité

$$I(x^i, y^j) = -\log_2 p_{XY}(i,j)$$

– entropie conjointe de deux variables aléatoires discrètes X et Y, l'espérance de l'information conjointe

$$H(X,Y) = -\sum_{i=1}^{L_X}\sum_{j=1}^{L_Y} p_{XY}(i,j)\log_2 p_{XY}(i,j)$$

[3]Par exemple $X(n)$ et $X(n+k)$. Dans ce cas, $L_X = L_Y$.

– information conditionnelle de l'événement $\{Y = y^j\}$ sachant que l'événement $\{X = x^i\}$ est réalisé, la quantité

$$I(y^j|x^i) = -\log_2 p_{Y|X}(j|i)$$

– entropie conditionnelle de la variable aléatoire discrète Y sachant que l'événement $\{X = x^i\}$ est réalisé

$$H(Y|x^i) = -\sum_{j=1}^{L_Y} p_{Y|X}(j|i) \log_2 p_{Y|X}(j|i)$$

– entropie conditionnelle de Y sachant X, l'espérance de l'entropie conditionnelle précédente, c'est-à-dire

$$H(Y|X) = \sum_{i=1}^{L_X} p_X(i) H(Y|x^i)$$

$$H(Y|X) = -\sum_{i=1}^{L_X} \sum_{j=1}^{L_Y} p_{XY}(i,j) \log_2 p_{Y|X}(j|i).$$

La relation

$$p_{XY}(i,j) = p_X(i) p_{Y|X}(j|i)$$

entraîne la relation suivante entre l'entropie $H(X)$, l'entropie conjointe $H(X,Y)$ et l'entropie conditionnelle $H(Y|X)$

$$H(X,Y) = H(X) + H(Y|X). \tag{4.6}$$

En effet

$$H(X,Y) = -\sum_{i=1}^{L_X} \sum_{j=1}^{L_Y} p_{XY}(i,j) \log_2 p_{XY}(i,j)$$

$$H(X,Y) = -\sum_{i=1}^{L_X} \sum_{j=1}^{L_Y} p_{XY}(i,j) \log_2 p_X(i) p_{Y|X}(j|i)$$

$$H(X,Y) = -\sum_{i=1}^{L_X} p_X(i) \log_2 p_X(i) - \sum_{i=1}^{L_X} \sum_{j=1}^{L_Y} p_{XY}(i,j) \log_2 p_{Y|X}(j|i).$$

4.3.2 Théorème du codage sans bruit d'une source discrète avec mémoire

Les définitions précédentes et la relation (4.6) se généralisent au cas de N variables aléatoires discrètes. Considérons le vecteur

$$\underline{X}_N = [X(mN) \cdots X(mN + N - 1)]^t$$

et la probabilité conjointe

$$P_{rob}\{X(mN) = x^{i_1}, \cdots, X(mN + N - 1) = x^{i_N}\} = p_{\underline{X}_N}(i_1, \cdots, i_N) = p_{\underline{X}_N}(\underline{i}).$$

On définit l'entropie de ce vecteur par

$$H(\underline{X}_N) = -\sum_{\underline{i}} p_{\underline{X}_N}(\underline{i}) \log_2 p_{\underline{X}_N}(\underline{i}).$$

On appelle débit entropique

$$\bar{H}(X) = \lim_{N \to \infty} \frac{1}{N} H(\underline{X}_N).$$

On montre qu'il est possible d'associer à une source un code uniquement décodable pourvu que le nombre de bits moyen par symbole soit supérieur ou égal au débit entropique.

Si la source est sans mémoire, on a

$$H(\underline{X}_N) = -N \sum_i p_X(i) \log_2 p_X(i) = NH(X).$$

Le débit entropique de cette source est donc égal à l'entropie de la source. De façon générale, on a l'inégalité

$$0 \leq \bar{H}(X) \leq H(X) \leq \log_2 L_X.$$

Tout ce développement démontre qu'il existe un code capable de représenter exactement la source à un débit égal au débit entropique mais n'explique pas toujours comment construire ce code.

On remarquera également qu'un codage efficace ne peut être réalisé qu'au dépend du délai de reconstruction.

4.3.3 Exemple d'une source markovienne

Généralités

Considérons le cas d'une source discrète $X(n)$ prenant ses valeurs dans $\{x^1 \cdots x^{L_X}\}$ et supposons connues toutes les probabilités conditionnelles que l'on ait l'événement $\{X(n) = x^i\}$ sachant qu'à l'instant précédent on avait l'événement $\{X(n-1) = x^j\}$. On suppose ces probabilités conditionnelles indépendantes de l'instant d'observation. On parle alors de source markovienne à l'ordre 1. Appelons

$$p(i|j) = p_{X(n)|X(n-1)}(i|j) = P_{rob}\{X(n) = x^i | X(n-1) = x^j\}.$$

Cette source peut être représentée par le diagramme d'état donné à la figure 4.4 où les différents

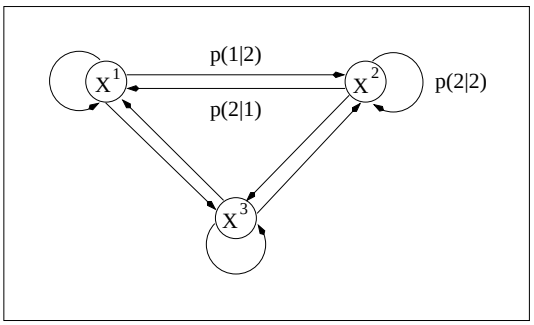

FIG. 4.4 – Diagramme d'état d'une source markovienne.

événements $\{X(n) = x^i\}$ sont les états possibles et les probabilités conditionnelles $p(i|j)$ sont les probabilités de transition. On note

$$p(i) = P_{rob}\{X(n) = x^i\}$$

la probabilité d'être dans l'état x^i. On suppose que $p(i)$ ne dépend pas de l'instant d'observation n. On ne se préoccupe donc pas d'un éventuel régime transitoire.

Si une probabilité $p(i|i)$ est importante, cela voudra dire que le symbole x^i apparaîtra vraisemblablement plusieurs fois de suite. On aura donc tout intérêt à coder ce symbole en spécifiant le nombre de répétition. On parle de codage par plage (*run-lenght coding*).

Exemple de la transmission de documents par télécopie

Il s'agit de transmettre par le réseau téléphonique public et de reproduire en noir et blanc un document dactylographié au format A4 (210×297 mm^2) en environ une minute à 4.8 kbit/s [22]. On donne simplement dans ce paragraphe quelques informations permettant de comprendre la méthode de codage effectuée. On se limite à la description du codage appelé codage unidimensionnel dans la recommandation. Il existe une deuxième méthode de codage, un codage bidimensionnel, qui tient compte d'une information relative à la ligne précédente mais cette méthode est trop particulière pour que sa description dans cet ouvrage soit intéressante.

La page A4 est lue de la gauche vers la droite et de haut en bas. La définition est de 4 lignes par mm et de 1728 éléments d'image (pixels) en noir et blanc par ligne. Il y a donc environ 2 Mbits à transmettre. Pendant une minute à 4.8 kbit/s, on transmet environ 300 kbits. Le taux de compression doit donc être voisin de 7.

On est dans le cas d'une source $X(n)$, la suite des pixels définie par le balayage sur les lignes, que l'on peut supposer markovienne à l'ordre 1. Cette source peut prendre deux états x^1 et x^2 correspondants respectivement à un pixel noir et à un pixel blanc. On remarque que toutes les probabilités sont entièrement déterminés par la simple connaissance des deux probabilités $p(2|1)$ et $p(1|2)$ puisque

$$\begin{aligned} p(1|1) &= 1 - p(2|1) \\ p(2|2) &= 1 - p(1|2) \end{aligned}$$

(il faut bien aller quelque part) et que

$$\begin{aligned} p(1) &= [1 - p(2|1)]p(1) + p(1|2)p(2) \\ p(2) &= 1 - p(1) \end{aligned}$$

soit

$$\begin{aligned} p(1) &= \frac{p(1|2)}{p(2|1) + p(1|2)} \\ p(2) &= \frac{p(2|1)}{p(2|1) + p(1|2)}. \end{aligned}$$

.

Si on est dans l'état x^1, l'entropie conditionnelle $H(X(n)|X(n-1) = x^1)$ représente le nombre de bits nécessaire pour coder son évolution à partir de l'état x^1. On a

$$H(X(n)|X(n-1) = x^1) = -p(1|1)\log_2 p(1|1) - p(2|1)\log_2 p(2|1) = f[p(2|1)]$$

avec $f(u) = -u\log_2 u - (1-u)\log_2(1-u)$ pour $0 \leq u \leq 1$ dont le graphe est visualisée figure 4.1. On obtient de même

$$H(X(n)|X(n-1) = x^2) = f[p(1|2)].$$

Le débit entropique est la moyenne (pondérée par la probabilité d'être dans l'état correspondant) des entropies conditionnelles. On obtient

$$H(X(n)|X(n-1)) = p(1)H(X(n)|X(n-1) = x^1) + p(2)H(X(n)|X(n-1) = x^2)$$

$$H(X(n)|X(n-1)) = g[p(1|2), p(2|1)].$$

En conclusion, sans connaissance des caractéristiques de la source, il faut utiliser 1 bit par symbole. Par contre, si les probabilités de transition sont faibles, le débit entropique est faible et il doit exister un code efficace.

Pour le construire, la méthode la plus simple consiste à

– scanner un grand nombre de feuilles écrites, dessinées, etc. pour constituer une base d'apprentissage,
– réaliser deux histogrammes sur le nombre de noirs (et blancs) consécutifs permettant une estimation des probabilités des événements : avoir un seul noir (blanc), deux noirs (blancs), ..., k noirs (blancs) successifs,
– appliquer l'algorithme de Huffman,
– comparer l'efficacité de cette méthode avec le calcul du débit entropique déduit d'une estimation de $p(1|2)$ et $p(2|1)$.

La recommandation parue en 1980 stipule que chaque mot de code représente effectivement le nombre de pixels blanc ou noir successifs. Il s'agit donc d'un codage par plage. Les séquences de noir et de blanc se produisent alternativement. Toute ligne commence par une séquence de blanc comportant éventuellement 0 élément et se termine par un mot de code particulier (EOL). Les mots de code sont à longueur variable. Il existe deux listes de mots de code distinctes pour les séquences correspondant au blanc et au noir. Les séquences d'une longueur de 0 à 63 éléments d'image correspondant au blanc sont codées avec les mots de code donnés table 4.6. Si une

0	00110101
1	000111
2	0111
3	1000
4	1011
5	1100
6	1110
7	1111
8	10011
9	10100
10	00111
11	01000
12	001000
13	000011
14	110100
15	110101
16	101010
17	101011
18	0100111
19	0001100
20	0001000
21	0010111
22	0000011
23	0000100
24	0101000
25	0101011
26	0010011
27	0100100
28	0011000
29	00000010
30	00000011
31	00011010

32	00011011
33	00010010
34	00010011
35	00010100
36	00010101
37	00010110
38	00010111
39	00101000
40	00101001
41	00101010
42	00101011
43	00101100
44	00101101
45	00000100
46	00000101
47	00001010
48	00001011
49	01010010
50	01010011
51	01010100
52	01010101
53	00100100
54	00100101
55	01011000
56	01011001
57	01011010
58	01011011
59	01001010
60	01001011
61	00110010
62	00110011
63	00110100

64	11011
128	10010
192	010111
256	0110111
320	00110110
384	00110111
448	01100100
512	01100101
576	01101000
640	01100111
704	011001100
768	011001101
832	011010010
896	011010011
960	011010100
1024	011010101
1088	011010110
1152	011010111
1216	011011000
1280	011011001
1344	011011010
1408	011011011
1472	010011000
1536	010011001
1600	010011010
1664	011000
1728	010011011
EOL	000000000001

TAB. 4.6 – Mots de code spécifiant la longueur des séquences correspondant au blanc lors de la transmission de documents par télécopie.

séquence a une longueur supérieure ou égale à 64 éléments, un premier mot de code spécifie dans quel intervalle il appartient, le second mot de code spécifie le reste de la division par 64. Il existe une deuxième table de mots de code spécifiant la longueur des séquences correspondant au noir

(table non fournie dans ce livre).

4.4 Quantificateur scalaire avec contrainte entropique

4.4.1 Introduction

On a donné au début du chapitre 1 une présentation très intuitive du quantificateur scalaire uniforme puis on s'est posé le problème de la détermination du quantificateur scalaire optimal lorsque le processus à quantifier $X(n)$ a une densité de probabilité marginale $p_X(x)$ quelconque. On rappelle la formule (1.1) donnant la puissance de l'erreur de quantification

$$\sigma_Q^2 = \sum_{k=1}^{L} \int_{x \in \Theta^k} (x - \hat{x}^k)^2 p_X(x) dx$$

en fonction de la partition $\{\Theta^1 \cdots \Theta^L\}$ de l'axe réel, des représentants $\{\hat{x}^1 \cdots \hat{x}^L\}$ et de la densité de probabilité $p_X(x)$ lorsque l'on choisit comme mesure de distorsion, l'erreur quadratique. La minimisation de σ_Q^2 relativement aux paramètres t^k définissant la frontière entre les partitions Θ^k et Θ^{k+1} et relativement aux représentants $\hat{x}^k$ ne fournit pas une solution analytique exploitable directement. On obtient simplement deux conditions nécessaires d'optimalité

$$t^k = \frac{\hat{x}^k + \hat{x}^{k+1}}{2}$$

$$\hat{x}^k = \mathbb{E}\{X | X \in \Theta^k\}.$$

Ces deux conditions sont exploitées dans l'algorithme de Lloyd-Max largement utilisé dans la pratique pour construire un quantificateur mais on ne peut pas assurer l'optimalité du quantificateur obtenu. On sait simplement que la suite des puissances obtenues par cet algorithme est une suite non-croissante ! On a vu également que, lorsque l'on suppose que le nombre de niveaux de quantification L est élevé, on obtient explicitement, contrairement au cas précédent, l'expression de la partition optimale uniquement en fonction de la densité de probabilité $p_X(x)$.

On introduit dans cette section un nouveau mode de quantification (scalaire) appelé quantification avec contrainte entropique. Dans la minimisation précédente permettant d'obtenir les deux conditions nécessaires d'optimalité, il s'agissait en fait d'une minimisation sous contrainte. La contrainte était que le nombre de niveaux de quantification L était fixe, égal à 2^b. On peut très bien chercher à minimiser σ_Q^2 relativement aux t^k et $\hat{x}^k$ sous la contrainte que l'entropie $H(\hat{X})$ de la sortie du quantificateur soit inférieure ou égale à b. Comme

$$H(\hat{X}) \leq \log_2 L = b$$

on peut espérer obtenir un quantificateur plus performant.

On se placera dans le cas où le nombre de niveaux de quantification L est élevé : la densité de probabilité peut être alors supposée constante dans l'intervalle $[t^{k-1}, t^k]$ et le représentant $\hat{x}^k$ peut être pris au milieu de cet intervalle.

4.4.2 Quantificateur de Lloyd-Max

On rappelle la formule (1.3) donnant la puissance de l'erreur de quantification

$$\sigma_Q^2 = \frac{1}{12} \sum_{k=1}^{L} p_X(\hat{x}^k) \Delta^3(k)$$

en fonction de $\Delta(k) = t^k - t^{k-1}$, la longueur de l'intervalle $[t^{k-1}, t^k]$.

Donnons une nouvelle justification de la formule (1.5) en rappelant d'abord qu'un quantificateur scalaire non-uniforme est équivalent à un quantificateur scalaire uniforme précédé d'une transformation non-linéaire et suivi de la transformation inverse. En appelant δ le pas de quantification du quantificateur uniforme, $f(x)$ la caractéristique et $g(x)$ la dérivée de $f(x)$, on a la relation

$$\frac{\delta}{\Delta(k)} \approx f'(\hat{x}^k) = g(\hat{x}^k). \tag{4.7}$$

La puissance de l'erreur de quantification s'écrit

$$\sigma_Q^2 = \frac{1}{12} \sum_{k=1}^{L} p_X(\hat{x}^k) \frac{\delta^2}{g^2(\hat{x}^k)} \Delta(k).$$

Par passage à la limite, on obtient

$$\sigma_Q^2 = \frac{\delta^2}{12} \int_{-\infty}^{+\infty} \frac{p_X(x)}{g^2(x)} dx.$$

Supposons, pour simplifier, que le support de $p_X(x)$ soit l'intervalle $[-A, +A]$ et que la densité de probabilité soit une fonction paire. Le pas de quantification δ du quantificateur uniforme est donné par

$$\delta = \frac{2f(A)}{L}.$$

La fonction $g(x)$ n'est pas quelconque. Elle doit vérifier la contrainte

$$\int_{-A}^{+A} g(x)dx = \int_{-A}^{A} f'(x)dx = 2f(A).$$

Le problème consiste donc à rechercher

$$\bar{g}(x) = \frac{g(x)}{2f(A)}$$

minimisant

$$\sigma_Q^2 = \frac{1}{12} \left(\int_{-A}^{+A} \frac{p_X(x)}{\bar{g}^2(x)} dx \right) 2^{-2b}$$

sous la contrainte

$$\int_{-A}^{+A} \bar{g}(x)dx = 1.$$

Utilisons l'inégalité de Hölder qui dit que

$$\int_{-A}^{+A} u(x)v(x)dx \leq \left(\int_{-A}^{+A} u^a(x)dx \right)^{1/a} \left(\int_{-A}^{+A} v^b(x)dx \right)^{1/b}$$

pourvu que a et b soient des entiers positifs vérifiant

$$\frac{1}{a} + \frac{1}{b} = 1.$$

L'inégalité est remplacée par une égalité lorsque $u^a(x)$ est proportionnel à $v^b(x)$. Choisissons

$$u(x) = \left(\frac{p_X(x)}{\bar{g}^2(x)} \right)^{1/3}$$

$$v(x) = \bar{g}^{2/3}(x)$$

et $a = 3$ et $b = 3/2$. On obtient

$$\int_{-A}^{+A} p_X^{1/3}(x)dx \leq \left(\int_{-A}^{+A} \frac{p_X(x)}{\bar{g}^2(x)}dx\right)^{1/3} \left(\int_{-A}^{+A} \bar{g}(x)dx\right)^{2/3}.$$

On a donc

$$\sigma_Q^2 \geq \frac{1}{12}\left(\int_{-A}^{+A} p_X^{1/3}(x)dx\right)^3 2^{-2b}$$

avec l'égalité si

$$\bar{g}(x) = \frac{p_X^{1/3}(x)}{\int_{-A}^{A} p_X^{1/3}(x)dx}.$$

On retrouve la formule (1.4) donnant la valeur minimale de la puissance de l'erreur de quantification.

4.4.3 Quantificateur avec contrainte entropique

Il s'agit de déterminer δ et $g(x)$ minimisant

$$\sigma_Q^2 = \frac{1}{12}\int_{-\infty}^{+\infty} [\frac{\delta}{g(x)}]^2 p_X(x)dx \tag{4.8}$$

en imposant une contrainte sur l'entropie $H(\hat{X})$ de la sortie du quantificateur

$$H(\hat{X}) \leq b.$$

Expression de l'entropie

L'entropie de la sortie du quantificateur a pour expression

$$H(\hat{X}) = -\sum_{k=1}^{L} P_{rob}(k)\log_2 P_{rob}(k)$$

avec

$$P_{rob}(k) = P_{rob}\{X \in [t^{k-1}, t^k]\} = p(\hat{x}^k)\Delta(k).$$

On a donc

$$H(\hat{X}) = -\sum_{k=1}^{L} p_X(\hat{x}^k)\Delta(k)\log_2 p_X(\hat{x}^k)\Delta(k)$$

$$H(\hat{X}) = -\sum_{k=1}^{L} p_X(\hat{x}^k)\Delta(k)\log_2 p_X(\hat{x}^k) - \sum_{k=1}^{L} p_X(\hat{x}^k)\Delta(k)\log_2 \Delta(k)$$

ou en exploitant la formule (4.7)

$$H(\hat{X}) = -\sum_{k=1}^{L} p_X(\hat{x}^k)\Delta(k)\log_2 p_X(\hat{x}^k) - \sum_{k=1}^{L} p_X(\hat{x}^k)\Delta(k)\log_2 \frac{\delta}{g(\hat{x}^k)}.$$

Par passage à la limite, on obtient

$$H(\hat{X}) = -\int_{-\infty}^{+\infty} p_X(x)\log_2 p_X(x)dx - \int_{-\infty}^{+\infty} p_X(x)\log_2 \frac{\delta}{g(x)}dx.$$

Le premier terme ne dépend que de la source. Il s'appelle l'entropie différentielle et on le note conventionnellement

$$h(X) = -\int_{-\infty}^{+\infty} p_X(x) \log_2 p_X(x) dx.$$

L'entropie différentielle caractérise la quantité d'information que possède une source continue sans mémoire. Elle partage la plupart des propriétés de l'entropie d'une source discrète sans mémoire excepté le fait qu'elle n'est pas nécessairement positive. Par exemple, pour une source présentant une distribution uniforme dans l'intervalle $[-A/2, A/2]$, on obtient

$$h(X) = \int_{-\infty}^{+\infty} \frac{1}{A} \log_2 A \; dx = \log_2 A.$$

Le signe de $h(X)$ dépend du support de $p_X(x)$. Si A est respectivement inférieur ou supérieur à 1, l'entropie différentielle sera négative ou positive.

Inégalité de Jensen

Cette inégalité est souvent utilisée en théorie de l'information. Elle dit que si X est une variable aléatoire sur l'intervalle $[a, b]$ et si $f(x)$ est une fonction convexe dans cet intervalle, alors

$$\mathbb{E}\{f(X)\} \geq f(\mathbb{E}\{X\}).$$

De plus, si $f(x)$ est strictement convexe alors l'égalité a lieu si et seulement si $X = \mathbb{E}\{X\}$ avec une probabilité égale à 1, c'est à dire si X est une constante.

Démontrons le premier résultat lorsque X est une variable aléatoire discrète prenant les valeurs x^1 et x^2 avec les probabilités $p_X(1)$ et $p_X(2)$. Comme une fonction convexe est une fonction vérifiant

$$\lambda f(x^1) + (1-\lambda) f(x^2) \geq f(\lambda x^1 + (1-\lambda) x^2)$$

$\forall x^1, x^2 \in [a, b]$ et $0 \leq \lambda \leq 1$, on peut écrire

$$p_X(1) f(x^1) + (1 - p_X(1)) f(x^2) \geq f[p_X(1) x^1 + (1 - p_X(1)) x^2]$$

c'est à dire

$$\mathbb{E}\{f(X)\} \geq f(\mathbb{E}\{X\}).$$

Pour une variable aléatoire X prenant ses valeurs dans $\{x^1 \cdots x^L\}$, il suffit de supposer que cette propriété est vraie à l'ordre $k-1$ et d'écrire

$$\sum_{i=1}^{k} p_X(i) f(x^i) = p_X(k) f(x^k) + (1 - p_X(k)) \sum_{i=1}^{k-1} \frac{p_X(i)}{1 - p_X(k)} f(x^i)$$

$$\sum_{i=1}^{k} p_X(i) f(x^i) \geq p_X(k) f(x^k) + (1 - p_X(k)) f(\sum_{i=1}^{k-1} \frac{p_X(i)}{1 - p_X(k)} x^i)$$

$$\sum_{i=1}^{k} p_X(i) f(x^i) \geq f(p_X(k) x^k + (1 - p_X(k)) f(\sum_{i=1}^{k-1} \frac{p_X(i)}{1 - p_X(k)} x^i)$$

$$\sum_{i=1}^{k} p_X(i) f(x^i) \geq f(\sum_{i=1}^{k} p_X(i) x^i).$$

Par passage à la limite, ce résultat se généralise au cas d'une variable aléatoire à valeurs continues.

Quantificateur optimal

L'équation (4.8) s'écrit

$$\sigma_Q^2 = \frac{1}{12}\mathbb{E}\{[\frac{\delta}{g(X)}]^2\}.$$

Puisque la fonction x^2 est une fonction convexe, on a

$$\sigma_Q^2 \geq \frac{1}{12}(\mathbb{E}\{\frac{\delta}{g(X)}\})^2$$

avec l'égalité si $g(x)$ prend une valeur constante notée g_0 sur tout le support de $p_X(x)$. Ceci veut dire que le meilleur quantificateur de la source continue $X(n)$ est tout simplement le quantificateur uniforme suivi d'un codage entropique. Le rapport δ/g_0 doit vérifier la contrainte

$$h(X) - \log_2 \frac{\delta}{g_0} \leq b$$

ce qui entraîne

$$\frac{\delta}{g_0} \geq 2^{h(X)-b}.$$

La puissance de l'erreur de quantification est donc minorée par

$$\sigma_Q^2 = \frac{1}{12}2^{2h(X)}2^{-2b}. \tag{4.9}$$

Source gaussienne

Donnons l'expression de l'entropie différentielle d'une source gaussienne. On a

$$h(X) = \int_{-\infty}^{+\infty} p_X(x)[\log_2 \sqrt{2\pi\sigma_X^2} + \frac{x^2}{2\sigma_X^2}\log_2 e]dx$$

$$h(X) = \log_2 \sqrt{2\pi\sigma_X^2}\int_{-\infty}^{+\infty} p_X(x)dx + \frac{\log_2 e}{2\sigma_X^2}\int_{-\infty}^{+\infty} p_X(x)x^2dx$$

$$h(X) = \log_2 \sqrt{2\pi\sigma_X^2} + \frac{\log_2 e}{2}$$

$$h(X) = \frac{1}{2}\log_2 2\pi e\sigma_X^2.$$

En reportant dans (4.9), on obtient

$$\boxed{\sigma_Q^2 = \frac{\pi e}{6}\sigma_X^2 2^{-2b}}.$$

La constante $\pi e/6$ est égale à 1,42. Le gain apporté par le quantificateur avec contrainte entropique relativement au quantificateur de Lloyd-Max est de

$$\frac{\sqrt{3}\pi/2}{\pi e/6} = \frac{3\sqrt{3}}{e} = 1,91$$

soit 2,81 dB.

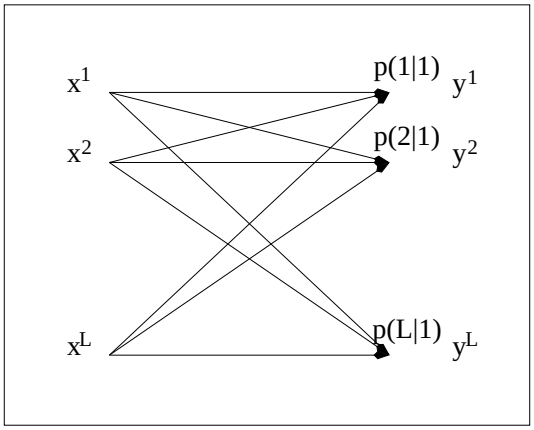

FIG. 4.5 – Canal bruité.

4.5 Capacité d'un canal discret sans mémoire

4.5.1 Introduction

Si un canal, recevant à l'instant n un symbole d'entrée $X(n)$ appartenant à l'alphabet $A_X = \{x^1 \cdots x^{L_X}\}$, fournit un symbole de sortie $Y(n)$ appartenant à l'alphabet $A_Y = \{y^1 \cdots y^{L_Y}\}$ et si le symbole de sortie ne dépend que du symbole d'entrée au même instant, on dira que le canal est discret et sans mémoire. On le représente par le schéma donné figure 4.5. Le canal peut être bruité. Il est donc naturel de chercher à caractériser l'application, à un symbole d'entrée correspond un symbole de sortie, par l'ensemble des probabilités conditionnelles

$$p_{Y|X}(j|i) = P_{rob}\{Y(n) = y^j | X(n) = x^i\}.$$

On supposera que les deux alphabets A_X et A_Y possèdent le même nombre de symboles. L'ensemble des probabilités conditionnelles peut être mis sous la forme d'une matrice carrée. Pour un canal peu bruité, cette matrice sera proche de la matrice identité.

La question que l'on se pose est la suivante : quelle est la quantité d'information que procure l'événement $\{Y(n) = y^j\}$ concernant l'événement $\{X(n) = x^i\}$?

4.5.2 Information mutuelle

L'information mutuelle des deux événements $\{X(n) = x^i\}$ et $\{Y(n) = y^j\}$ est la quantité

$$I(x^i; y^j) = -\log_2 \frac{p_X(i) p_Y(j)}{p_{XY}(i,j)}$$

$$I(x^i; y^j) = -\log_2 \frac{p_Y(j)}{p_{Y|X}(j|i)} = -\log_2 \frac{p_X(i)}{p_{X|Y}(i|j)}.$$

L'information mutuelle[4] est égale à zéro si l'événement $\{Y(n) = y^j\}$ est indépendant de l'événement $\{X(n) = x^i\}$. Elle est maximale si $p_{Y|X}(j|i) = 1$, c'est-à-dire si le canal est sans erreur pour cette paire. Dans ce cas, elle est égale à l'information propre $I(x^i)$. Elle représente donc la quantité d'information que fournit le symbole reçu y^j sur le symbole émis x^i.

[4]On observera soigneusement les différents signes de ponctuation dans les notations. La virgule étant traditionnellement associée à des probabilités conjointes, on utilise un point-virgule pour noter une information mutuelle.

L'information mutuelle moyenne est l'espérance de $I(x^i;y^j)$. Elle s'écrit

$$I(X;Y) = \sum_{i=1}^{L_X}\sum_{j=1}^{L_Y} p_{XY}(i,j) I(x^i;y^j)$$

$$I(X;Y) = -\sum_{i=1}^{L_X}\sum_{j=1}^{L_Y} p_{XY}(i,j) \log_2 \frac{p_X(i)p_Y(j)}{p_{XY}(i,j)}. \tag{4.10}$$

Donnons quelques propriétés de $I(X;Y)$. Remarquons tout d'abord que l'information mutuelle moyenne est l'entropie relative définie par l'équation (4.3) entre la probabilité conjointe et le produit des probabilités marginales

$$I(X;Y) = D(p_{XY}||p_X p_Y).$$

On en déduit que $I(X;Y)$ est non-négatif et vaut 0 si et seulement si les deux variables aléatoires X et Y sont indépendantes. L'information mutuelle moyenne $I(X;Y)$ mesure la dissimilitude entre la probabilité conjointe et ce qu'elle serait si les deux variables aléatoires étaient indépendantes. Elle mesure la quantité d'information que les deux variables aléatoires s'apportent mutuellement.

Comme l'équation (4.10) s'écrit

$$I(X;Y) = -\sum_{i=1}^{L_X}\sum_{j=1}^{L_Y} p_{XY}(i,j) \log_2 \frac{p_X(i)}{p_{X|Y}(i|j)}$$

$$I(X;Y) = -\sum_{i=1}^{L_X} p_X(i)\log_2 p_X(i) + \sum_{i=1}^{L_X}\sum_{j=1}^{L_Y} p_{XY}(i,j)\log_2 p_{X|Y}(i|j)$$

on obtient

$$I(X;Y) = H(X) - H(X|Y).$$

L'information mutuelle moyenne mesure donc la réduction d'incertitude sur X due à la connaissance de Y. L'entropie conditionnelle $H(X|Y)$ peut être considérée comme l'incertitude moyenne concernant le symbole émis par la source après que le symbole produit au récepteur ait été spécifié. Pour un canal peu bruité, l'entropie conditionnelle est presque nulle. L'information mutuelle est alors maximale. Elle est pratiquement égale à l'entropie de la source. L'information mutuelle moyenne caractérise le transfert d'information.

En développant l'équation (4.10), on obtient directement

$$I(X;Y) = H(X) + H(Y) - H(X,Y).$$

Lorsque $Y = X$, on a

$$I(X;X) = H(X).$$

Donnons l'expression de l'information mutuelle moyenne en mettant en évidence les probabilités conditionnelles $p_{Y|X}(j|i)$ qui caractérisent le canal et les probabilités $p_X(i)$ associées aux symboles de la source

$$I(X;Y) = -\sum_{i=1}^{L_X}\sum_{j=1}^{L_Y} p_X(i)p_{Y|X}(j|i) \log_2 \frac{p_Y(j)}{p_{Y|X}(j|i)}.$$

Comme

$$p_Y(j) = \sum_{k=1}^{L_X} p_X(k) p_{Y|X}(j|k)$$

on obtient

$$I(X;Y) = -\sum_{i=1}^{L_X}\sum_{j=1}^{L_Y} p_X(i)p_{Y|X}(j|i)\log_2 \frac{\sum_{k=1}^{L_X} p_X(k)p_{Y|X}(j|k)}{p_{Y|X}(j|i)}.$$

4.5.3 Théorème de codage pour un canal bruité

On appelle capacité d'un canal le maximum de l'information mutuelle moyenne sur toutes les distributions possibles de la source

$$C = \max_{p_X(i)} I(X;Y).$$

La capacité d'un canal est la plus grande quantité d'information qui peut être transmise à travers lui. Comme $I(X;Y) \geq 0$, $I(X;Y) = H(X) - H(X|Y)$ et que $H(X|Y) \geq 0$, on en déduit

$$C \leq \log_2 L.$$

La source est caractérisée par son débit entropique $\bar{H}(X)$. Le canal est caractérisé par sa capacité C. On montre que si

$$\bar{H}(X) \leq C$$

alors une transmission avec une probabilité d'erreur aussi faible que l'on veut est possible. On montre également que si

$$\bar{H}(X) > C$$

alors une transmission avec une probabilité d'erreur aussi faible que l'on veut devient impossible.

4.5.4 Exemple : canal binaire symétrique

Le canal est schématisé à la figure 4.6. Il est caractérisé par la matrice

$$\begin{bmatrix} p_{Y|X}(1|1) & p_{Y|X}(2|1) \\ p_{Y|X}(1|2) & p_{Y|X}(2|2) \end{bmatrix} = \begin{bmatrix} 1-p_e & p_e \\ p_e & 1-p_e \end{bmatrix}.$$

Calculons la capacité de ce canal. Il s'agit de maximiser l'information mutuelle moyenne en

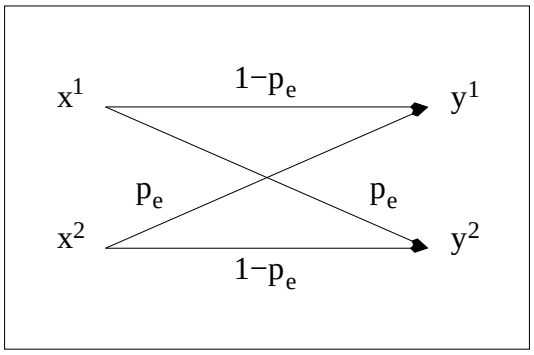

FIG. 4.6 – Canal binaire symétrique.

fonction de la probabilité $p = p_X(1)$. Utilisons la formule

$$I(X;Y) = H(Y) - H(Y|X).$$

Comme

$$H(Y|X) = pH(Y|x^1) + (1-p)H(Y|x^2)$$

et que

$$H(Y|x^1) = H(Y|x^2) = -(1-p_e)\log_2(1-p_e) - p_e \log_2 p_e$$

$$H(Y|x^1) = H(Y|x^2) = f(p_e)$$

on remarque que

$$H(Y|X) = pf(p_e) + (1-p)f(p_e) = f(p_e)$$

est indépendant de p. D'autre part, comme

$$H(Y) = f[p_Y(1)] = f[p(1-p_e) + (1-p)p_e]$$

et que la fonction $f(x)$ atteint son maximum pour $x = 1/2$ ce qui entraîne $p = 1/2$, on obtient $H(Y) = 1$. La capacité du canal binaire symétrique est donc égale à

$$C = 1 - f(p_e).$$

Lorsque le canal est non-bruité, la capacité est égale à 1. Lorsque $p_e = 1/2$, la capacité est nulle. Généralement le calcul de la capacité d'un canal est beaucoup plus difficile que dans ce cas là.

4.6 Codage d'une source discrète avec un critère de fidélité

4.6.1 Problème

Le théorème précédent indique que si la capacité du canal est insuffisante compte tenu du débit entropique de la source, aucune transmission n'est possible avec une probabilité d'erreur aussi faible que l'on veut mais c'est, en fait, le cas pratique qui nous intéresse. Au prix d'une certaine distorsion, on désire comprimer un signal. Est-il possible, dans ce cas, d'évaluer les performances du système ? Une fonction que l'on appelle fonction débit-distorsion permet de déterminer la plus petite distorsion tolérable lorsque $\bar{H}(X) > C$. Cette théorie montre l'existence de cette fonction mais ne fournit pas une méthode permettant de construire un code ayant ces performances.

Au lieu d'optimiser sur tous les codes déterministes ayant une structure donnée en respectant une contrainte sur le débit de transmission, on s'intéresse, comme dans tout ce chapitre, à des codes caractérisés par des fonctions aléatoires et on impose une contrainte sur le débit d'information.

4.6.2 Fonction débit-distorsion

Considérons une source discrète caractérisée par l'alphabet $A_X = \{x^1 \cdots x^{L_X}\}$. On rappelle qu'entre l'entropie $H(X)$ de la source, si l'on ne tient pas compte de sa mémoire éventuelle, son débit entropique $\bar{H}(X)$ et le nombre d'éléments L_X constituant l'alphabet A_X existent les inégalités

$$0 \leq \bar{H}(X) \leq H(X) \leq \log_2 L_X.$$

On n'examine ici que le cas le plus simple, à savoir le codage d'une source discrète sans mémoire. Introduisons un nouvel alphabet $A_{\hat{X}} = \{\hat{x}^1 \cdots \hat{x}^{L_{\hat{X}}}\}$ que l'on appellera alphabet de reproduction. L'opération de codage peut être représentée par le même schéma que celui de la figure 4.5 où on remplace y^k par $\hat{x}^k$. On caractérise chaque branche par les probabilités de transition

$$p_{\hat{X}|X}(j|i) = P_{rob}\{\hat{X}(n) = \hat{x}^j | X(n) = x^i\}.$$

Il faut, en plus, associer à chaque branche une mesure de distorsion que l'on supposera, pour simplifier, ne comportant pas d'effet de mémoire (ou d'anticipation). On parle alors de mesure

de distorsion par lettre et on la note $d(x^i, \hat{x}^j)$. Rappelons l'expression de l'information mutuelle moyenne

$$I(X;\hat{X}) = -\sum_{i=1}^{L_X}\sum_{j=1}^{L_{\hat{X}}} p_X(i)p_{\hat{X}|X}(j|i)\log_2 \frac{\sum_{k=1}^{L_X} p_X(k)p_{\hat{X}|X}(j|k)}{p_{\hat{X}|X}(j|i)}.$$

On cherche l'alphabet de reproduction optimal. Cela veut dire que les degrés de liberté sont, dans ce problème, les probabilités conditionnelles. Les probabilités $p_X(i)$ sont imposées par la source. Pour mettre ceci en évidence, on note l'information mutuelle moyenne

$$I(p_{\hat{X}|X}) = I(X;\hat{X}).$$

La distorsion moyenne a pour expression

$$\bar{d}(p_{\hat{X}|X}) = \sum_{i=1}^{L_X}\sum_{j=1}^{L_{\hat{X}}} p_{X\hat{X}}(i,j)d(x^i,\hat{x}^j)$$

$$\bar{d}(p_{\hat{X}|X}) = \sum_{i=1}^{L_X}\sum_{j=1}^{L_{\hat{X}}} p_X(i)p_{\hat{X}|X}(j|i)d(x^i,\hat{x}^j).$$

On appelle fonction débit-distorsion

$$b(D) = \min_{p_{\hat{X}|X}\in Q} I(p_{\hat{X}|X})$$

où Q est l'ensemble de toutes les probabilités conditionnelles vérifiant

$$\bar{d}(p_{\hat{X}|X}) \leq D.$$

La fonction débit-distorsion ne dépend que de la source.

Pour généraliser ce résultat à des sources avec mémoire, il faudrait regrouper N symboles d'entrée, former le vecteur

$$X_N = [x(mN)\cdots x(mN+N-1)]^t$$

et définir une suite de fonctions $b_N(D)$. On montre que

$$b_N(D) \geq b_{N+1}(D).$$

Si l'on veut une distorsion nulle, le codage doit se faire sans perte. On a $b(0) = H(X)$. Si D augmente, l'ensemble Q augmente également ce qui entraîne que la fonction débit-distorsion est une fonction non-croissante. Plus précisément, on montre que c'est une fonction continue, convexe, strictement décroissante. Si l'on accepte une distorsion supérieure ou égale à la puissance de la source (dans le cas d'une mesure de distorsion quadratique), alors il n'est même plus nécessaire de chercher à la coder. On a $b(\sigma_X^2) = 0$.

4.6.3 Théorèmes

Théorème du codage de source

Connaissant la fonction débit-distorsion $b(D)$, on peut montrer que quel que soit $D \geq 0$ et quel que soit $\epsilon > 0$, il existe un quantificateur vectoriel composé de L vecteurs de dimension N vérifiant

$$\frac{1}{N}\log_2 L < B(D) + \epsilon$$

et entraînant une distorsion moyenne inférieure ou égale à $D + \epsilon$.

Codage combiné source-canal

Le théorème de codage pour un canal bruité dit que si la capacité du canal est suffisante, c'est-à-dire supérieure ou égale au débit entropique de la source, il n'y a pas de problème. Si cette capacité est insuffisante, le résultat nouveau est que l'on peut d'abord réaliser une compression en acceptant une distorsion telle que $b(D) = C$ puis effectuer un codage canal avec une probabilité d'erreur aussi faible que l'on veut.

4.6.4 Cas particulier : mesure de distorsion quadratique

Borne inférieure de Shannon pour une source sans mémoire

Pour une mesure de distorsion quadratique, on montre que la fonction débit-distorsion pour une source non-gaussienne sans mémoire de puissance σ_X^2 est minorée par

$$b(D) \geq h(X) - \frac{1}{2}\log_2 2\pi e D.$$

Cette inégalité permet de donner la borne inférieure de la puissance de l'erreur de quantification pour une source gaussienne sans mémoire

$$b = \frac{1}{2}\log_2 2\pi e\sigma_X^2 - \frac{1}{2}\log_2 2\pi e\sigma_Q^2 = \frac{1}{2}\log_2 \frac{\sigma_X^2}{\sigma_Q^2}$$

soit

$$\sigma_Q^2 = \sigma_X^2\ 2^{-2b}. \tag{4.11}$$

Le quantificateur scalaire avec contrainte entropique a une puissance de l'erreur de quantification égale à $\pi e/6 = 1,42$ fois cette limite. On dit que le quantificateur scalaire avec contrainte entropique est à

$$\frac{1}{2}\log_2 \frac{\pi e}{6} = 0,25 \quad \text{bit}$$

de la limite théorique. La courbe $b(D) + 0,25$ est parfois appelée l'asymptote de Gish-Pierce.

Ce résultat est intéressant car il montre que toute source sans mémoire et de puissance σ_X^2 a une fonction débit-distorsion en dessous de celle d'une source gaussienne et de même puissance lorsque la mesure de distrosion est l'erreur quadratique. La source gaussienne est donc la source la plus difficile à reproduire.

Source avec mémoire

De la même façon que pour une source discrète avec mémoire, on généralise sans trop de difficulté le résultat précédent. On appelle débit entropique différentiel d'une source continue, la limite

$$\bar{h}(X) = \lim_{N\to\infty} \frac{1}{N} h(X_N)$$

avec

$$h(X_N) = -\int_{R^N} p_{X_N}(x)\log_2 p_{X_N}(x)dx$$

où $p_{X_N}(x)$ est la densité de probabilité conjointe du vecteur aléatoire composé de N symboles successifs. Pour une source gaussienne centrée, de puissance σ_X^2 et de matrice d'autocorrélation normalisée Γ_X, la densité de probabilité conjointe a pour expression

$$p_{X_N}(x) = \frac{1}{(2\pi\sigma_X^2)^{N/2}\sqrt{det\ \Gamma_X}} e^{-(x^t\Gamma_X^{-1}x)/2\sigma_X^2}.$$

On a donc

$$\begin{aligned} h(X_N) &= \frac{1}{2}\int_{-\infty}^{+\infty} p_X(x)\log_2(2\pi\sigma_X^2)^N det\ \Gamma_X dx \\ &\quad + \frac{\log_2 e}{2\sigma_X^2}\int_{-\infty}^{+\infty} p_X(x) x^t \Gamma_X^{-1} x dx \\ h(X_N) &= \frac{1}{2}\log_2(2\pi\sigma_X^2)^N det\ \Gamma_X + \frac{\log_2 e}{2\sigma_X^2}\mathbb{E}\{x^t\Gamma_X^{-1}x\}. \end{aligned} \tag{4.12}$$

Comme on démontre que

$$\mathbb{E}\{x^t\Gamma_X^{-1}x\} = N\sigma_X^2$$

on obtient

$$h(X_N) = \frac{1}{2}\log_2(2\pi e\sigma_X^2)^N det\ \Gamma_X$$

$$\bar{h}(X) = \lim_{N\to\infty}\frac{1}{2}\log_2 2\pi e\sigma_X^2 (det\ \Gamma_X)^{1/N}$$

$$\bar{h}(X) = \frac{1}{2}\log_2\frac{2\pi e\sigma_X^2}{G_p(\infty)}$$

à cause de la relation (1.12). On peut en déduire que la borne inférieure de la puissance de l'erreur de quantification pour une source gaussienne avec mémoire est égale à

$$\sigma_Q^2 = \frac{\sigma_X^2}{G_p(\infty)}\ 2^{-2b}. \tag{4.13}$$

4.6.5 Généralisation

Considérons la formule (4.11). On en déduit directement le nombre de bits minimum nécessaire pour quantifier une source avec un rapport signal sur bruit donné

$$b \geq \frac{1}{2}\log_2\frac{\sigma_X^2}{\sigma_Q^2}. \tag{4.14}$$

La formule (4.14) admet une première généralisation comme on vient de le voir. On obtient la formule (4.13). Dans la pratique, on a vu qu'il existe plusieurs méthodes permettant d'exploiter la corrélation (filtre blanchissant, quantification vectorielle, transformation ou bancs de filtres ...). La valeur asymptotique $G_p(\infty)$ du gain de prédiction est uniquement fonction de la densité spectrale $S_X(f)$ de la source

$$G_p(\infty) = \frac{\int_{-1/2}^{1/2} S_X(f)df}{e^{\int_{-1/2}^{1/2}\log_e S_X(f)df}}. \tag{4.15}$$

Les relations (4.13) et (4.15) permettent d'obtenir le nombre de bits minimum nécessaire pour quantifier une source corrélée

$$b \geq \frac{1}{2}\int_{-1/2}^{+1/2}\log_2\frac{S_X(f)}{\sigma_Q^2}df. \tag{4.16}$$

Une deuxième généralisation est nécessaire car, dans le cadre du codage de signaux audiofréquence, on verra que ce n'est pas la minimisation de la puissance du bruit de quantification qui est le problème essentiel. On cherchera quel est le débit nécessaire et suffisant pour que la densité spectrale de puissance du bruit soit inférieure à une densité spectrale de puissance limite fournie par un modèle d'audition. On montre [18] que l'équation (4.16) se généralise sous la forme suivante

$$\boxed{b \geq \frac{1}{2}\int_{-1/2}^{+1/2}\max[0, \log_2\frac{S_X(f)}{S_Q(f)}]\ df}. \tag{4.17}$$

Intuitivement, cette formule se déduit directement de (4.16) ou de (4.14) en observant qu'il est inutile de chercher à quantifier le signal dans les bandes de fréquences où $S_X(f) \leq S_Q(f)$ et que, dans les bandes de fréquence "élémentaires" $[f, f + df[$ vérifiant $S_X(f) > S_Q(f)$, la "densité de bits" nécessaire est donnée par

$$\frac{db}{df} \geq \frac{1}{2} \log_2 c(1) \ \frac{S_X(f)}{S_Q(f)}.$$

La théorie indique donc que, pour coder un signal, connaissant la densité spectrale de puissance du signal et du bruit tolérable, il faut déterminer les bandes de fréquence $[f_k, f_k^{'}]$ telles que $S_X(f) \geq S_Q(f)$ puis allouer les ressources binaires disponibles en fonction du rapport $S_X(f)/S_Q(f)$.

Deuxième partie

Application aux signaux audio

Chapitre 5

Introduction aux signaux audio

5.1 Caractéristiques des signaux de parole

Le signal de parole est un signal très complexe possédant de nombreuses caractéristiques. On se contentera ici de présenter quelques propriétés significatives en compression.

La figure 5.1 montre un signal de parole de durée une demi-seconde prononcé par un locuteur féminin dans le domaine temporel (à gauche) et dans le domaine fréquentiel (à droite). On voit sur le tracé de gauche que ce signal n'est évidemment pas stationnaire mais qu'il peut être considéré comme localement stationnaire pendant des durées de l'ordre de quelques dizaines de ms. En codage de la parole, il est tout à fait standard de choisir des fenêtres d'analyse de 20 ms.

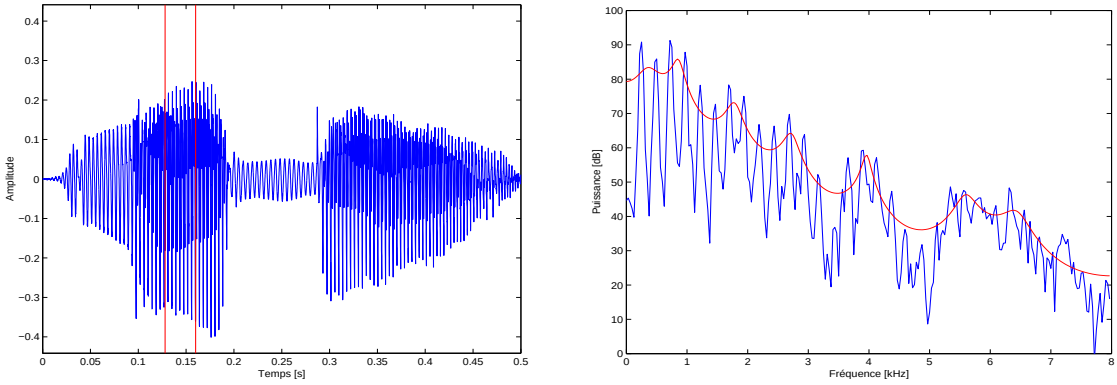

FIG. 5.1 – Exemple d'un signal de parole (La bas ...) dans le domaine temporel (à gauche) et dans le domaine fréquentiel (à droite). Les échantillons qui ont permis la détermination des deux estimations spectrales sont compris entre les deux barres verticales du tracé de gauche.

On distingue ensuite différents types de sons : des sons voisés, des sons non-voisés[1], des plosives. La compression des sons voisés et non-voisés peut être réalisée dans de bonnes conditions comme on le verra par la suite. Il n'en sera pas de même, par contre, pour les plosives et pour les transitions entre phonèmes.

La troisième caractéristique, très importante comme on le verra par la suite également, est l'existence d'un modèle de production simple et efficace. Pour s'en convaincre examinons le tracé de droite de la figure 5.1 donnant deux estimations spectrales. La première estimation est réalisée en calculant un périodogramme, c'est-à-dire en prenant le module au carré de la transformée de Fourier discrète de N échantillons. On admettra que ce calcul procure une bonne approximation

[1] Pas d'exemple de ce type sur ce tracé.

de la véritable (mais inaccessible) densité spectrale de puissance. Appelons $\hat{S}_1(f)$ cette densité spectrale. Elle est représentée par la courbe présentant de nombreux pics. La deuxième estimation $\hat{S}_2(f)$ utilise un modèle autorégressif, c'est-à-dire admet que le signal analysé est le résultat du filtrage d'un bruit blanc par un filtre ne comportant que des pôles. La courbe $\hat{S}_2(f)$ est beaucoup plus lisse. Elle correspond à l'enveloppe spectrale. Une observation détaillée de ces courbes montre que $\hat{S}_1(f)$ peut être vu, en première approximation, comme la multiplication de $\hat{S}_2(f)$ par un ensemble d'impulsions de même puissance régulièrement espacées[2]. Cela veut dire qu'il parait raisonnable de modéliser un son voisé par le filtrage d'un peigne de Dirac par un filtre autorégressif. Pour des sons non-voisés, des test d'écoute montrent qu'il est raisonnable d'utiliser le même type de filtre à la condition de remplacer le peigne de Dirac par un bruit blanc.

5.2 Caractéristiques des signaux de musique

Le tracé de gauche de la figure 5.2 montre un signal de violon d'une durée d'une demi-seconde. Comme pour le signal de parole, on remarque des variations de puissance instantanée

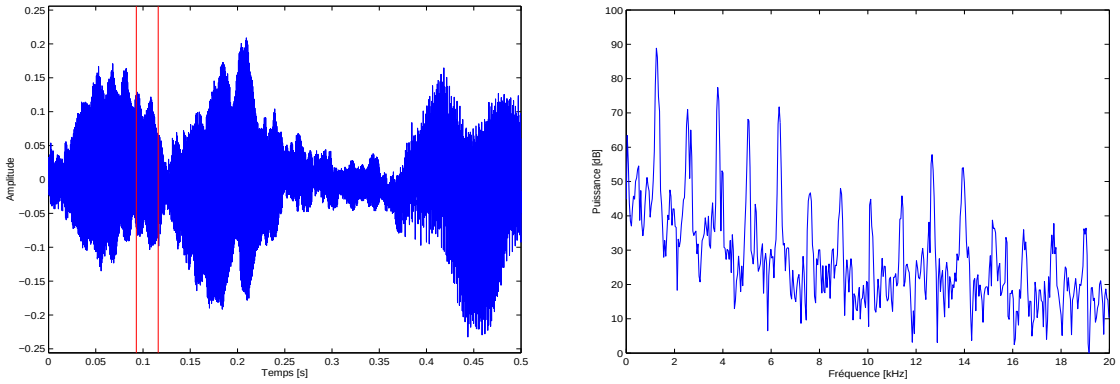

FIG. 5.2 – Exemple d'un signal de violon dans le domaine temporel (à gauche) et dans le domaine fréquentiel (à droite).

importantes. Elles peuvent être considérables, jusqu'à 90 dB, et pas uniquement pour des signaux très percussifs. Cela peut être le cas par exemple dans une symphonie de Mahler entre des passages pianissimo et des passages fortissimo ! La représentation spectrale sur le tracé de droite, entre 0 et 20 kHz (ce signal est échantillonné à 44.1 kHz), traduit le timbre de l'instrument. Les *partiels* qui sont les composantes spectrales prédominantes peuvent être reliées harmoniquement ou pas. Dans le cas du violon, on peut observer que c'est approximativement le cas. Ce qu'il faut surtout noter c'est qu'il n'existe plus de modèle de production simple à mettre en œuvre. Les sons musicaux sont produits de façons trop différentes.

5.3 Normes et recommandations

Pour plus de détails, on se reportera à l'excellent article[3] [23].

5.3.1 Signal de parole en bande téléphonique

On a classé les différentes normalisations et recommandations en fonction des applications.

[2] Tout au moins jusqu'à 2 kHz environ. L'écart entre les impulsions qui est, dans cet exemple, environ égal à 200 Hz, est appelé la fréquence fondamentale, la fréquence de *pitch*.

[3] mais qui aurait besoin d'être actualisé !

Réseau téléphonique public

Plusieurs "recommandations" ont été définies sous l'égide de l'Union Internationale des Télécommunications, secteur des Télécommunications (UIT-T) pour le réseau téléphonique public ces trente dernières années.

- Depuis 1972, la recommandation G.711 précise un codage par modulation par impulsions codées (MIC ou PCM pour pulse code modulation) correspondant à un débit de 64 kbit/s : l'amplitude des échantillons est simplement quantifiée sur 8 bits après une compression de type non linéaire. La plupart des réseaux téléphoniques commutés du monde entier utilisent ce codeur.
- Depuis 1984, la recommandation G.721 définit un codage MIC différentiel adaptatif (MICDA ou ADPCM) correspondant à un débit de 32 kbit/s : on ne quantifie plus directement l'amplitude de l'échantillon mais la différence entre l'amplitude et une valeur prédite déterminée par un filtrage de type adaptatif. Ce codeur est utilisé pour la concentration de voies pour les transmissions par cables sous-marins ou par satellites. Il est aussi utilisé pour la norme européenne de téléphone sans cordon DECT (Digital Enhanced Cordless Telecommunications).
- Pour les débits 16/24/32/40 kbit/s, il existe une version multidébit (G.726) ou à codes imbriqués (G.727) du codeur précédent.
- Un codeur à 16 kbit/s basé sur des techniques de modélisation et de quantification vectorielle, a été sélectionné par l'UIT-T en 1991. Cette recommandation G.728 est également appelée LD-CELP (Low Delay Code Excited Linear Predictive coder) mettant en évidence que c'est un codeur de type CELP (long développement par la suite) et qu'il présente un faible délai de reconstruction, propriété particulièrement importante pour un échange téléphonique.
- Une compétition internationale a sélectionné en 1995 un codeur à 8 kbit/s associant le codeur ACELP (Algebraic Code Excited Linear Predictive coder) étudié à l'Université de Sherbrooke (Canada) et dans les laboratoires de France Telecom à Lannion et le codeur de NTT. Il s'agit du codeur G.729.
- Enfin un codeur à 6.3 kbit/s, le codeur G.723.1, spécifie la partie son du visiophone.

Communication avec les mobiles

Les communications avec les mobiles ont connu un grand développement ces vingt dernières années. La nature du canal de transmission, une liaison radio, réclame d'économiser au maximum la largeur de bande du signal transmis pour permettre un grand nombre d'utilisateurs.

- Depuis 1989, il existe une norme européenne dite GSM 06.10 (ou 6.20 ?) (Groupe Spécial Mobile) sélectionnée par le *European Telecommunications Standard Institute* (ETSI). Une norme concernant l'Amérique du Nord appelée IS 54 a été définie en 1990 par le *Telecommunication Industry Association* (TIA). L'ouverture du service Itineris par France Telecom a eu lieu le 1er Juillet 1992. Cette première génération était basée sur les techniques d'accès multiples par division du temps (TDMA) et sur les codeurs de source[4], RPE-LTP (Regular Pulse Excitation - Long Term Prediction) en Europe à 13 kbit/s, VSELP (Vector Sum Excited Linear Predictive coder) en Amérique du Nord à 8 kbit/s.
- Une nouvelle norme (GSM 06.60) a été définie en 1996 visant à améliorer la qualité du codeur GSM. Il s'agit du codeur EFR (Enhanced Full-Rate) [24]. En absence d'erreurs, on peut dire que sa qualité est équivalente à celle du G.726 à 32 kbit/s.
- Depuis 1999 existe un nouveau codeur (GSM 06.90) réalisant le partage du débit entre

[4]Tous les débits indiqués dans ce paragraphe correspondent au codage de source. Il faut rajouter approximativement le même débit pour le codage du canal, par exemple 9,8 kbit/s dans le cas du GSM, le débit total étant de 22,8 kbit/s.

codage de source et codage de canal dépendant des conditions de propagation. Il est à débit variable (codeur AMR pour Adaptive Multi Rate) entre 4,75 et 12,2 kbit/s (en réalité entre 11,4 et 22,8 kbit/s).
- De nouvelles standardisations (3GPP pour 3^{rd} Generation Partnership Project) apparaissent visant l'émergence des systèmes de communication avec les mobiles dits de troisième génération *Universal Mobile Telephone System* (UMTS). Une collaboration UIT/ETSI a permis la définition du codeur G.722.2 qui est un codeur de parole en bande élargie.

Autres applications

Dans d'autres applications, par exemple pour des communications sécurisées entre organismes inter-gouvernementaux, pour des applications militaires (norme OTAN), pour des communications avec des mobiles par satellite (MSAT et INMARSAT), on réclame un débit faible au prix d'une dégradation importante du signal. Le département de la Défense américain (US DoD) a standardisé en 1991 un codeur à 4,8 kbit/s, appelé FS 1016, pour remplacer l'ancien standard fédéral FS 1015 ou LPC10 de qualité très médiocre défini en 1976.

5.3.2 Signal de parole en bande élargie

L'intérêt de transmettre un signal de parole en bande élargie (f_e=16 kHz) est d'obtenir un signal de parole reconstitué plus net et plus intelligible que dans le cas de la bande téléphonique. Les applications sont les conférences audiovisuelles, le visiophone, la téléphonie sur haut-parleurs.
- Une norme UIT-T à 64 kbit/s dite G.722 a été normalisée en 1986. Ce codeur est basé sur un codage en deux sous-bandes contenant un codeur MICDA (norme G.721) dans chaque sous-bande. Pour pouvoir transmettre simultanément des données sur le même canal, un débit réduit à 56 kbit/s est également possible.
- Un nouveau codeur (G.722.1) a été défini en 1999 pour des débits de 24 et 32 kbit/s.
- Une nouvelle normalisation UIT-T en 2002, le codeur G.722.2, a été au préalable accepté par l'organisme de normalisation 3GPP sous la dénomination de AMR-WB. Une description très complète est disponible dans [25].

5.3.3 Signal de musique en bande Hi-Fi

Le format de référence pour le signal de musique est habituellement celui du disque compact : le signal est échantillonné à 44.1 kHz puis quantifié scalairement sur 16 bits. Le débit correspondant est alors de 705 kbit/s (en mono, 1.4 Mbit/s en stéréo).

Concernant le support, on peut noter qu'il est fragile, il peut être rayé, et que, par conséquent, les bits doivent être protégés. Deux étages de codes correcteurs de type Reed Solomon entraînant un débit de 2.9 Mbit/s ont été introduits. Un CD peut être interprété comme un canal de transmission ayant un débit de 4.3 Mbit/s.

On peut noter également qu'il existe d'autres supports [26] : le "Super Audio CD" (SACD) et le DVD-Audio. Le premier est le résultat de l'échantillonnage du signal à 2.82 MHz sur 1 bit (modulation sigma/delta), le second correspond à une fréquence d'échantillonnage de 48, 96 ou 192 kHz et à une quantification sur 24 bits. Ces deux supports, actuellement en compétition, permettent de stocker de 2 à 6 canaux. Ils entraînent approximativement la même qualité (une amélioration de la dynamique en puissance de l'ordre d'une vingtaine de dB relativement à celle du CD) et la même capacité (de l'ordre de 5 Go). Qu'en sera-t-il des nouveaux supports annoncés, le HD-DVD et le Blu-Ray Disc ?

MPEG-1

La première application nécessitant une opération de compression a été celle de la diffusion de signaux audio sous forme numérique[5] (DAB). Ces études ont abouti en 1992 à la partie audio de la norme internationale ISO/CEI 11172 [27], plus connue sous le nom de MPEG-1 Audio. Des deux codeurs "en finale", le codeur MUSICAM et le codeur ASPEC, c'est le premier qui a été sélectionné. Le codeur est composé de trois "couches" (layers) de qualité équivalente pour des débits de 192, 128 et 96 kbit/s mais de complexité croissante. Le célèbre format MP3 est en réalité le MPEG-1 layer 3. Bien qu'il corresponde à la complexité la plus importante, cette complexité est considérée maintenant comme très raisonnable.

MPEG-2

Dans le domaine de l'audio, il n'y a pas une très grande différence entre le codeur MPEG-1 et le codeur MPEG-2 normalisé en 1994 (ISO/IEC 13818) pour des applications de type TVHD, DVD, minidisque[6]. Le second est une extension multi-voies (5.1) du premier. Il introduit beaucoup plus de souplesse dans le choix des fréquences d'échantillonnage, des débits. Ce nouveau codeur a la propriété d'être "backward compatible" c'est à dire qu'un décodeur MPEG-1 a la possibilité d'interpréter un flux MPEG-2 ce qui est plus inhabituel que d'être "forward compatible" lorsque un décodeur MPEG-2 peut aussi décoder un flux MPEG-1. Cette propriété est importante dans un réseau de diffusion car s'il est facile de basculer d'un codeur à un autre à l'émetteur par contre on ne peut pas demander à des millions de téléspectateurs de changer brutalement de décodeurs. Cette compatibilité arrière est pénalisante en terme de performance du système de compression si bien que des nouvelles études ont été lancées se libérant de cette contrainte et ont abouti en 1997 à une nouvelle normalisation [28]. Il s'agit du codeur AAC (Audio Advanced Coder) qui assure une qualité "transparente" avec des taux de compression plus élevés. Il est réputé comme étant transparent au débit de 384 kbit/s dans la configuration 5.1, ce qui correspond approximativement à une transparence à 64 kbit/s en mono. C'est le codeur le plus performant à l'heure actuelle, c'est le codeur qui est utilisé dans les iPod.

Signalons qu'il existe d'autres codeurs "propriétaires", par exemple le Dolby AC3 paru en 1995, le Sony ATRC3 pour le minidisque (qui a disparu), etc.

MPEG-4

MPEG-4 est un standard beaucoup plus ambitieux. Il a pour vocation de représenter des sons (parole et musique) d'origine naturelle (issus d'un microphone) ou d'origine synthétique (fabriqués par une machine). Il a aussi pour vocation de définir des "objets sonores" susceptibles d'être manipulés de façon à former des "scènes sonores". La première normalisation a eu lieu en 1998 [29]. Elle a été ensuite complétée en 1999. Cette normalisation a subi plusieurs révisions depuis. La présentation qui suit est issue du "Draft International Standard" datant de 2005 [30].

Pour des sons **d'origine naturelle**, MPEG-4 ne cherche pas à fournir des codeurs assurant nécessairement la transparence ou même proches de la transparence comme pour les standards MPEG-1 et MPEG-2. Il définit une famille de codeurs de 2 à plusieurs centaines de kbit/s assurant la meilleure qualité possible pour un débit donné. En dessous de 64 kbit/s et *a fortiori* pour des débits très faibles, il ne faut pas rêver : la qualité peut être raisonnable, on parle alors de qualité

[5] On remarquera que cette application, étudiée dès le milieu des années 80, n'a toujours pas véritablement été exploitée !

[6] Les professionnels de l'audio, au début des années 80, semblent avoir été très réticents à l'introduction de traitements numériques des signaux de musique, réticence qui a progressivement cédée à cause de la souplesse et des performances de ces techniques. Ils, en particulier les gens de télévision, se sont ralliés aux techniques de compression à la fin des années 80.

"intermédiaire", ou même médiocre. MPEG-4 définit aussi une famille de codeurs hiérarchiques ("scalables") pour avoir la possibilité de modifier le débit en cas de congestion dans un réseau. Le train binaire est construit en plusieurs couches de telle sorte que le décodeur puisse traiter tout ou une partie du train binaire (au détriment de la qualité).

MPEG-4 fournit une boite à outils regroupant plusieurs algorithmes de compression :

- De 2 à 24 kbit/s pour des **signaux de parole** en bande téléphonique ou en bande élargie, deux techniques de codage sont utilisées.
  - Le codeur MPEG-4 HVXC (Harmonic Vector eXcitation Coding) opère à des débits compris entre 2 et 4 kbit/s. Il traite des signaux de parole en bande téléphonique. Il accepte des manipulations de type "time-stretching" (modification de la durée d'un enregistrement sans modification du pitch) permettant, par exemple, de consulter de larges bases de données de parole de façon rapide.
  - Le codeur MPEG-4 CELP prend le relais à partir de 4 kbit/s. Il permet de traiter des signaux de parole soit en bande téléphonique, soit en bande élargie. Pour des signaux de parole en bande téléphonique, c'est un codeur CELP standard très comparable au codeur UIT-T G.729. Pour des signaux de parole en bande élargie, un premier codage basé sur l'exploitation d'un modèle de production comme précédemment traite la bande de fréquences $[0-4]$ kHz. Un second codage basé sur l'exploitation d'un modèle d'audition comme dans un codeur de type MPEG-1 ou MPEG-2 se charge de la bande $[4-7]$ kHz.
- De 6 à plusieurs centaines de kbit/s pour des **signaux de musique** monophoniques, stéréophoniques ou multicanaux (généralement au format 5.1), le codeur MPEG-4 AAC est presque une recopie du codeur MPEG-2 AAC. Pour des débits supérieurs à 64 kbit/s par canal on sait que ce codeur vérifie le critère de transparence. Pour des débits inférieurs, il cherche la meilleure qualité pour un débit donné. Il existe plusieurs variantes de ce codeur :
  - la version "Low Delay" pour des communications interactives,
  - une variante dite BSAC (Bit Slice Arithmetic Coding) pour avoir une scalabilité de "granularité" très fine (1kbit/s),
  - une version TWIN-VQ (Transform Weighted INterleave-Vector Quantization) assurant de meilleures performances que le codeur AAC entre 6 et 16 kbit/s.

  Pour gagner en débit sans toucher à la qualité, un outil récent a été rajouté. Il s'agit de l'outil SBR (Spectral Band Replication) qui est le résultat de plusieurs travaux comme par exemple [31]. Dans cette méthode, la contribution hautes fréquences du signal de musique est reconstruite à partir du spectre basses fréquences avec très peu d'informations additionnelles. Ainsi on peut obtenir un débit de 24 kbit/s en utilisant 22 kbit/s pour les basses fréquences (le codeur AAC codant un signal sous échantillonné par un facteur 2) et 2 kbit/s pour reconstruire les hautes fréquences. Les performances du codeur AAC sont alors nettement améliorées à faible débit. Ce codeur est un candidat pour plusieurs normes : Digital Radio Mondiale[7], 3GPP, DVB.

  Dans le cadre des signaux de musique il faut aussi rajouter dans la liste précédente les codeurs paramétriques.
  - Le codeur MPEG-4 HILN (Harmonic and Individual Line plus Noise coding) code du signal de musique dans la gamme des débits 4 à 16 kbit/s. Il utilise une représentation du signal basée sur des sinusoïdes, des groupes de sinusoïdes reliées harmoniquement et des composantes de bruit. Les paramètres propres à chaque "objet" sont codés individuellement. Cela procure une grande souplesse à la restitution. Comme pour le codeur de parole HVXC il accepte des manipulations de type "time-stretching".
  - Cette idée est généralisée dans le cas du codeur MPEG-4 SSC (SinuSoidal Coding) qui décompose le signal en davantage d'objets : des sinusoïdes, des transitoires, du bruit et également une "image spatiale" dans le cas de signaux stéréo ou même multicanaux

[7] DRM à ne pas confondre avec Digital Right Management au cœur de MPEG-21 !

comme on le verra au chapitre 8. Il est capable de coder un signal pleine bande à débit réduit avec une très bonne qualité.

Lorsque le codeur MPEG-4 AAC est utilisé avec les outils SBR et "Parametric stereo" il prend le nom de MPEG-4 AAC+ ou HE-AACv2 (High Efficiency) [32].

Pour terminer il faut citer le codeur MPEG-4 SLS (Scalable Lossless coding). Il s'agit de codage sans perte ou scalable du débit offert par le codeur AAC jusqu'au sans perte avec une granularité assez fine. On n'est donc plus dans la gamme de débit précédente. Les débits concernés sont compris entre 64 et 705 kbit/s pour un signal monophonique. On notera que le codage sans perte n'assure pas un taux de compression bien élevé, de l'ordre de 2. La limite inférieure pour le débit est donc de l'ordre de 300 kbit/s pour du sans perte.

Pour des sons **d'origine synthétique**, on trouve :

- Un algorithme de synthèse de la parole (synthèse "Text-to-Speech") qui est un outil assez basique pour des applications multi-média. Les informations de parole transmises sont soit des éléments de l'alphabet phonétique international, soit du texte écrit dans n'importe quel langage.
- Un langage pour engendrer de la musique. Il s'agit du langage SAOL (Structured Audio Orchestra Language). MPEG-4 permet d'exploiter le format MIDI.
- MPEG-4 standardize la façon de décrire une scène. Il définit l'endroit dans un système de coordonnés où se trouve (se déplace) un objet sonore (navigation dans une scène). Il décrit également la façon dont est modifiée l'apparence de chaque objet. Il peut s'agir de modifications prosodiques pour de la parole ou de réverbération et de spatialisation pour de la musique.

Le développement précédent, axé essentiellement dans une optique compression, laisse sous silence de nombreuses potentialités de la norme MPEG-4. Pour juste suggérer ces potentialités, on reprendra un exemple cité dans [30].

Supposons que, dans une application particulière, on désire transmettre avec une bonne qualité le son créé par une personne parlant dans un environnement réverbérant avec un fond musical stéréo. Dans une approche traditionnelle, il suffirait d'utiliser un codeur audio fonctionnant à 32 kbit/s par canal par exemple. Avec MPEG-4 on peut représenter le signal global comme la réunion de plusieurs objets : le son prononcé par une personne passant au travers d'un réverbérateur auquel on additionne un extrait de musique produit de façon synthétique. On transmet la voix en utilisant l'outil CELP à un débit de 16 kbit/s, on synthétise la musique en utilisant l'outil SA à un débit de 2 kbit/s et on rajoute 1 ou 2 kbit/s pour décrire l'aspect stéréo et le réverbérateur. En conclusion, à qualité équivalente, l'approche MPEG-4 orientée objets coûte moins de 20 kbit/s alors qu'une approche plus traditionnelle demanderait 64 kbit/s. De plus la personne qui écoute au récepteur peut vouloir n'écouter que la personne seule ce que permet seulement l'approche MPEG-4.

MPEG-7, MPEG-21

Les travaux de MPEG-7 ne traitent pas des problèmes de compression. Il s'agit d'indexation. MPEG-21 axe ses travaux sur des problèmes liés à la "sécurité" de contenus multi-média. Les techniques employées sont, par exemple, les techniques de tatouage.

5.3.4 Evaluation de la qualité

La qualité des signaux audio reconstruits (parole ou musique) ne peut pas être appréciée à l'aide de critères objectifs du type rapport signal à bruit. Comme on le verra plus particulièrement lorsque l'on étudiera les codeurs de musique, à puissances de bruit (due à la compression)

équivalentes, certains codeurs entraînent une bonne qualité et d'autres une mauvaise. La forme spectrale du bruit joue un rôle très important dans sa perception. On est donc obligé de se contenter de tests subjectifs (des écoutes) mais ces tests subjectifs sont dits formels car les protocoles sont définis très précisément.

Pour des codeurs de parole de qualité intrinsèquement médiocre, on réalise des tests d'intelligibilité : méthode de jugement par catégories absolues ACR (Absolute Category Rating), par catégories de dégradation DCR (Degradation Category Rating), etc.

Pour les codeurs audio, on veut généralement qu'ils soient de très bonne qualité. On exige alors la "transparence". Pour pouvoir comparer des codeurs entre eux, on emploie la méthode dite "doublement aveugle à triple stimulus et référence dissimulée". Il s'agit de la recommandation UIT-R BS.1116. On fait écouter à de jeunes (au delà de 30 ans, l'oreille se dégrade) musiciens (on réclame des oreilles exercées) des enregistrements courts (entre 5 et 10 secondes) de morceaux de musique sélectionnés (existence d'une base de données MPEG). Chaque enregistrement est répété 3 fois. On offre deux possibilités : soit la séquence A/B/A soit la séquence A/A/B où A représente le signal original et B le signal codé/reconstruit. La première réponse réclamée est la suivante : B se trouve-t-il en 2ème ou en 3ème position ? On réclame aussi une opinion (une note entre 0 et 5) sur B : le bruit dû à la compression est-il totalement inaudible (5), très légèrement gênant (4), un peu gênant (3), gênant (2), mauvais (1) ou très mauvais (0) ? Un traitement statistique est ensuite réalisé permettant une comparaison "objective" entre codeurs.

Pour des codeurs audio de débits compris entre 20 et 64 kbits/s où on est obligé de se contenter d'une qualité "intermédiaire" ou "acceptable", c'est la méthode MUSHRA (MUlti Stimulus test with Hidden Reference and Anchor) qui est généralement employée. Il s'agit de la recommandation UIT-R BS.1534-1.

Notons qu'il existe tout de même des tests objectifs qui peuvent donner des résultats significatifs. On peut par exemple citer l'algorithme PEAK au coeur de la recommandation UIT-R BS.1387-1 [33].

Chapitre 6

Codeurs de signaux de parole

6.1 Les codeurs MIC et MICDA

Pour du signal de parole en bande téléphonique la réduction de bande relativement à de la parole naturelle est importante. Une perte de qualité significative en résulte. Une quantification sur 16 bits qui assure, associée à une fréquence d'échantillonnage de 44.1 kHz, une qualité "compact-disc" n'est donc pas nécessaire. On admettra qu'une quantification sur 12 bits est "homogène" à une fréquence d'échantillonnage de 8 kHz. Le débit de référence pour du signal de parole en bande téléphonique est donc de l'ordre de 100 kbit/s.

Le codeur MIC (PCM) UIT-T G.711 à 64 kbit/s réalise une quantification scalaire non uniforme comme cela a été présenté (très rapidement) à la sous-section 1.2.3. Une simple quantification scalaire est mal adaptée à des signaux présentant des variations de puissance instantanée importantes. On montre que pour maintenir un rapport signal à bruit à peu près constant, la transformation non-linéaire adaptée est de nature logarithmique. Cette transformation est généralement approximée par des segments de droite. On parle de "loi A" en Europe et de "loi μ aux États-Unis.

Le schéma de principe du codeur MICDA (ADPCM) UIT-T G.726 à 32 kbit/s est celui de la figure 1.4 correspondant à un quantificateur scalaire prédictif en boucle fermée. Il y a plusieurs modèles de prédiction linéaire possibles. Le codeur G.726 utilise un modèle ARMA(2,6) où les deux coefficients spécifiques de la partie AR et les six coefficients spécifiques de la partie MA sont actualisés à la cadence échantillon. L'estimation de ces coefficients est réalisée par une méthode du type gradient mais il s'agit en réalité d'une méthode du gradient simplifiée pour gagner en rapidité de calcul. Le détail des opérations n'a pas de valeur pédagogique et ne sera donc pas présenté ici.

Comme la puissance de l'erreur de prédiction est approximativement proportionnelle à celle du signal, ce codeur introduit un facteur d'échelle $g(n)$ telle que $2^{-g(n)}y(n)$ soit de puissance constante. En réalité, $g(n)$ est estimé récursivement à l'aide d'une équation récurrente de la forme $g(n) = (1-\alpha)\ g(n-1) + \alpha\ w(n)$ où le paramètre $\alpha \approx 10^{-6}$ caractérise la vitesse d'adaptation de $g(n)$.

6.2 Le codeur LPC10 à 2.4 kbit/s

Ce codeur ne présente plus aucun intérêt pratique. Par contre, il présente un grand intérêt pédagogique car il est à la base des codeurs de parole actuels.

Considérons le schéma de la figure 6.1 où $x(n)$ est le signal de parole original, $y(n)$ le signal en sortie du filtre d'"analyse", $\hat{y}(n)$ l'entrée du filtre de "synthèse" et $\hat{x}(n)$ le signal de parole

reconstruit. Le codeur LPC10 calcule les coefficients du filtre

$$A(z) = 1 + a_1 z^{-1} + \cdots a_P z^{-P}$$

à partir du signal original puis détermine l'entrée du filtre de synthèse de telle sorte que la qualité du signal reconstruit soit la meilleure possible tout en respectant la contrainte de débit (2.4 kbit/s). Il exploite des fenêtres d'analyse d'une vingtaine de ms en supposant le signal "localement stationnaire" dans chacune de ces fenêtres. On notera par la suite

$$\underline{x} = [x(0) \cdots x(N-1)]^t$$

les $N = 160$ échantillons (pour du signal de parole échantillonné à $f_e = 8$ kHz) dans chacune de ces fenêtres.

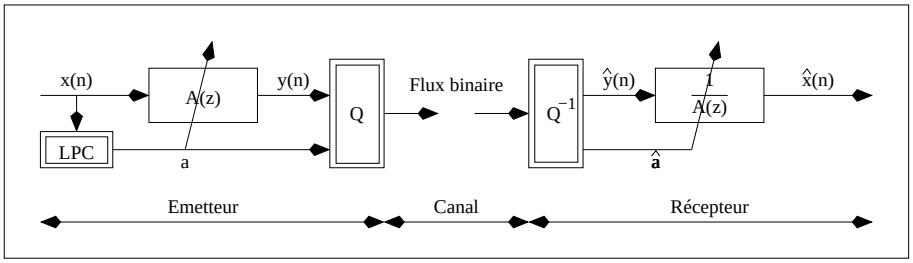

FIG. 6.1 – Schéma très général d'un codeur de parole.

6.2.1 Détermination des coefficients du filtre

On s'appuie sur la théorie de la prédiction linéaire. On rappelle que la puissance de l'erreur de prédiction s'exprime en fonction des coefficients du filtre prédicteur et de la fonction d'autocovariance $r_X(k)$ du processus $X(n)$ sous la forme

$$\sigma_Y^2 = \sigma_X^2 + 2\underline{r}^t\underline{a} + \underline{a}^t\mathbb{R}\underline{a}$$

où

$$\underline{a} = \begin{bmatrix} a_1 \\ \vdots \\ a_P \end{bmatrix} \quad \underline{r} = \begin{bmatrix} r_X(1) \\ \vdots \\ r_X(P) \end{bmatrix} \quad \mathbb{R} = \begin{bmatrix} r_X(0) & \cdots & r_X(P-1) \\ \vdots & \ddots & \vdots \\ r_X(P-1) & \cdots & r_X(0) \end{bmatrix}.$$

En écrivant que la dérivée de σ_Y^2 relativement au vecteur $\underline{a}$ est égale au vecteur nul, on obtient les "équations normales"

$$\mathbb{R}\underline{a}^{opt} = -\underline{r}$$

et la puissance minimale

$$(\sigma_Y^2)^{min} = \sigma_X^2 + \underline{r}^t\underline{a}^{opt}.$$

Dans la pratique, il faut réaliser une estimation de la fonction d'autocovariance. On remplace une moyenne statistique $r_X(k) = \mathbb{E}\{X(n)X(n-k)\}$ par une moyenne temporelle

$$\hat{r}_X(k) = \frac{1}{N}\sum_{n=k}^{N-1} x(n)x(n-k).$$

On rappelle que le choix particulier des bornes de la sommation qui impose que l'on utilise strictement les données uniquement dans l'intervalle $[0, N-1]$ est dû au fait que l'on obtient une propriété pratique très importante : on assure ainsi la stabilité du filtre de synthèse. Une fois déterminés, ces coefficients doivent être quantifiés puis les mots de code transmis ce qui permet de reconstruire au récepteur les coefficients du filtre de synthèse.

6.2.2 Cas des sons non voisés

La théorie de la prédiction linéaire permet d'affirmer que si $x(n)$ peut être considéré comme la réalisation d'un processus aléatoire AR d'ordre P_0, alors il existe un filtre de fonction de transfert $A(z)$ totalement blanchissant dès que son ordre P devient supérieur ou égal à P_0. Dans ce cas, on peut écrire que la densité spectrale de puissance de l'erreur de prédiction est égale à

$$S_Y(f) = \sigma_Y^2.$$

Comme on sait aussi que

$$S_Y(f) = |A(f)|^2 S_X(f),$$

on en déduit que

$$S_X(f) = \frac{\sigma_Y^2}{|A(f)|^2}.$$

Supposons que l'on choisisse comme entrée du filtre de synthèse une réalisation quelconque d'un bruit blanc[1] de puissance $\sigma_{\hat{Y}}^2 = \sigma_Y^2$. On voit alors que l'on a la propriété suivante au niveau des densités spectrales de puissance

$$S_{\hat{X}}(f) = \frac{\sigma_Y^2}{|A(f)|^2} = S_X(f).$$

On reconstruit un signal qui a la même distribution de la puissance en fonction de la fréquence mais les formes d'onde sont différentes. Comme l'oreille est relativement insensible à des modifications de la phase, on reconstruit par ce mécanisme un signal qui est perçu approximativement identique au signal original. On appelle ce type de codeur un "vocodeur".

Tout ce raisonnement est valable en supposant que le signal de parole puisse être considéré comme la réalisation d'un processus aléatoire AR. Cette hypothèse est assez réaliste pour des sons non-voisés uniquement.

6.2.3 Cas des sons voisés

Les tracés de la figure 6.2 sont relatifs à un son voisé où l'on montre aussi bien dans le domaine temporel (à gauche) que dans le domaine fréquentiel (à droite) le signal original $x(n)$ et l'erreur de prédiction $y(n)$. Le filtre $A(z)$ n'est manifestement pas totalement blanchissant. Il reste dans le signal $y(n)$ une périodicité assez marquée, visible aussi bien dans le domaine temporel que dans le domaine fréquentiel. Pendant une durée de 32 ms, on a approximativement 7.5 périodes. La fréquence fondamentale est donc de l'ordre de $f_0 \approx 7.5/0.032 \approx 250$ Hz. On observe bien dans le domaine fréquentiel un spectre de raies avec une fréquence fondamentale de 250 Hz (correspondant à un locuteur féminin) et les différents harmoniques.

Le problème qui se pose maintenant est de trouver un modèle $\hat{y}(n)$ pour $y(n)$ qui permette par filtrage d'obtenir $S_{\hat{X}}(f) \approx S_X(f)$ et qui soit très économe en débit. Un peigne de la forme

$$\hat{y}(n) = \alpha \sum_{m=-\infty}^{+\infty} \lambda(n - mT_0 + \varphi)$$

est un bon candidat. Dans cette expression, $\lambda(n)$ est le symbole de Kronecker qui vaut 1 si $n = 0$, 0 sinon, $T_0 = f_e/f_0$ est la période fondamentale exprimée en nombre d'échantillons et φ une valeur appartenant à $\{0, \cdots, T_0 - 1\}$ traduisant notre incertitude sur la phase. Le signal $\hat{y}(n)$ peut être alors interprété comme la réalisation d'un processus aléatoire $\hat{Y}(n)$ dont il est intéressant de déterminer les propriétés.

[1] en exploitant la fonction matlab randn par exemple

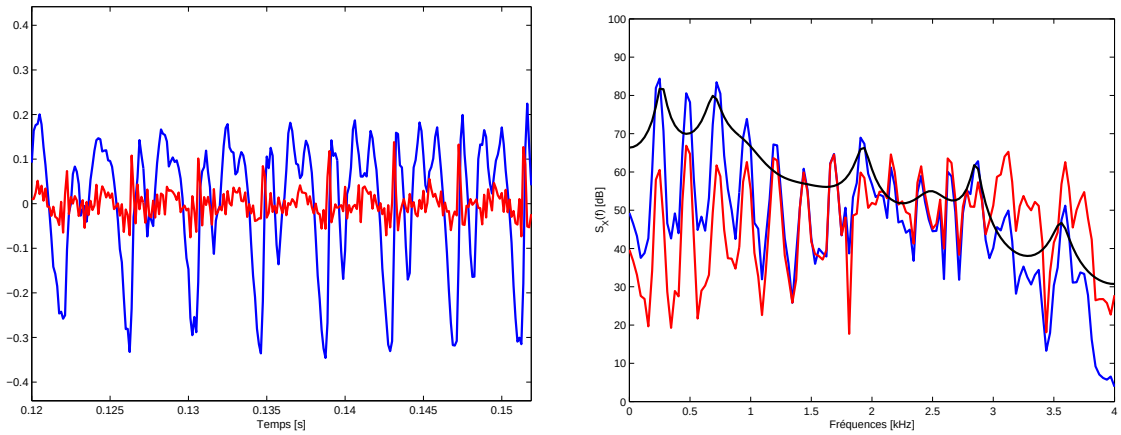

FIG. 6.2 – Cas de sons voisés.

On remarque que la moyenne a pour expression

$$\mathbb{E}\{\hat{Y}(n)\} = \alpha \sum_{\varphi=0}^{T_0-1} \frac{1}{T_0} \sum_{m=-\infty}^{+\infty} \lambda(n - mT_0 + \varphi) = \frac{\alpha}{T_0}$$

et que la fonction d'autocorrélation[2] qui est donnée par

$$r_{\hat{Y}}(k,n) = \mathbb{E}\{\hat{Y}(n)\hat{Y}(n-k)\} = \alpha^2 \, \mathbb{E}\{ \sum_{m=-\infty}^{+\infty} \lambda(n - mT_0 + \varphi) \sum_{l=-\infty}^{+\infty} \lambda(n - k - lT_0 + \varphi)\}$$

vaut

$$r_{\hat{Y}}(k,n) = \alpha^2 \, \mathbb{E}\{ \sum_{m=-\infty}^{+\infty} \lambda(n - mT_0 + \varphi)\} = \frac{\alpha^2}{T_0}$$

si k est un multiple de T_0, 0 sinon. La moyenne et la fonction d'autocorrélation étant indépendantes de l'instant d'observation n, le processus $\hat{Y}(n)$ est stationnaire de moyenne $1/T_0$ et de fonction d'autocorrélation

$$r_{\hat{Y}}(k) = \frac{\alpha^2}{T_0} \sum_{m=-\infty}^{+\infty} \lambda(k - mT_0).$$

La densité spectrale de puissance $S_{\hat{Y}}(f)$ est la transformée de Fourier à temps discret de $r_{\hat{Y}}(k)$. On a donc

$$S_{\hat{Y}}(f) = \frac{\alpha^2}{T_0} \sum_{k=-\infty}^{+\infty} \sum_{m=-\infty}^{+\infty} \lambda(k - mT_0) \exp(-j2\pi fk) = \frac{\alpha^2}{T_0} \sum_{m=-\infty}^{+\infty} \exp(-j2\pi fmT_0)$$

ou de façon équivalente

$$S_{\hat{Y}}(f) = \alpha^2 \sum_{m=-\infty}^{+\infty} \delta(f - \frac{m}{T_0})$$

en se rappelant qu'un train d'impulsions de dirac admet une sorte de développement en série de Fourier et que l'on a donc la formule générale

$$\sum_{n=-\infty}^{+\infty} \delta(t - nT) = \frac{1}{T} \sum_{n=-\infty}^{+\infty} e^{j2\pi \frac{n}{T} t}.$$

[2]Son expression est plus simple que celle de la fonction d'autocovariance.

Comme le tracé de droite de la figure 6.2 a été obtenu à partir d'un nombre fini de données observées, on "voit" sur ce tracé le lobe principal de la transformée de Fourier à temps discret de la fenêtre de pondération. Dans cette simulation, on a pris une fenêtre de Hamming de durée $2N$. Le lobe principal a pour largeur $4f_e/2N$ ce qui correspond à 100 Hz, valeur cohérente avec celle observée sur le tracé.

Le filtrage de $\hat{y}(n)$ par le filtre dont le module au carré de la réponse en fréquence est représenté en noir sur la figure 6.2 donnera bien un signal dont la densité spectrale de puissance se superposera avec le tracé en bleu de cette figure.

6.2.4 Détermination sons voisés/sons non voisés

Le codeur LPC10 doit déterminer s'il s'agit d'un son voisé ou non voisé dans la fenêtre d'analyse courante. Cette détermination se fait en réalisant au préalable une estimation de la fonction d'autocorrélation normalisée de l'erreur de prédiction $y(n)$ pour de nombreuses valeurs de k. Si cette fonction décroît vers 0 rapidement (comparaison à un seuil), on en déduit que le son est non voisé. Dans le cas d'un signal voisé, cette fonction est presque périodique de période T_0 ce qui fournit un estimateur de la période fondamentale. Dans la pratique, il faut prendre quelques précautions car un algorithme rudimentaire a tendance à déterminer comme fréquence fondamentale, le 2ème ou même le 3ème harmonique ... Dans la pratique, il faudra assurer des transitions pas trop brutales lorsque l'on passera d'une fenêtre d'analyse à la suivante. Il faudra, en particulier, faire en sorte que le choix de φ (information non codée) entraîne une transition douce entre les peignes successifs.

6.2.5 Contrainte de débit

Toutes les 20 ms, on transmet les paramètres suivants.

- Les coefficients du filtre $A(z)$. Dans le codeur LPC10, $P = 10$. En prenant comme ordre de grandeur 3 ou 4 bits pour coder un coefficient, on obtient un débit de l'ordre de 1.8 kbit/s.
- La puissance σ_Y^2 dans le cas d'un son non voisé ou de façon équivalente le coefficient α dans le cas d'un son voisé ce qui coûte $50 \times 6 \approx 300$ bit/s (pour couvrir 50 dB par pas de 1 dB).
- La distinction voisé/non voisé soit 50 bits/s.
- Enfin la période fondamentale soit $50 \times \log_2(T_0^{max} - T_0^{min}) = 350$ bits/s.

En sommant toutes ces contributions, on obtient un débit global de 2.5 kbit/s !

6.3 Le codeur CELP

6.3.1 Introduction

La plupart des codeurs de parole en bande téléphonique, spécialement dans la gamme des débits de 4.8 à 16 kbit/s, sont des codeurs de type "CELP" (Code Excited Linear Predictive coder").

Le principe de ces codeurs a été introduit par B. Atal au début des années 80. La première idée de base de B. Atal [34] a été de proposer un nouveau modèle pour l'excitation du filtre de synthèse de la forme

$$\hat{y}(n) = \sum_{k=1}^{K} g_k \lambda(n - n_k)$$

et de calculer les paramètres g_k et n_k de ce modèle en minimisant le critère

$$\sum_{n=0}^{N-1} [x(n) - \hat{x}(n)]^2.$$

Comparativement à la détermination de l'entrée du filtre de synthèse du codeur LPC10 dans le cas de sons voisés, il y a deux innovations. Il ne s'agit plus d'un peigne car la position des "impulsions" n'est plus régulière et il ne s'agit plus de rendre $S_{\hat{X}}(f) \approx S_X(f)$ mais bien de rendre $\hat{x}(n) \approx x(n)$. On parle de modélisation "par la synthèse". A l'émetteur on cherche à construire explicitement ce qui sera effectivement réalisé au récepteur. La deuxième idée de B. Atal trois ans plus tard [35] a été d'exploiter la quantification vectorielle qui venait d'être introduite. L'idée consistait à proposer comme candidat possible pour l'entrée du filtre de synthèse un vecteur de dimension N choisi dans un dictionnaire prédéfini comme le montre le schéma de la figure 6.3 et un gain g. Ce schéma de principe montre que le problème s'exprime simplement. Connaissant

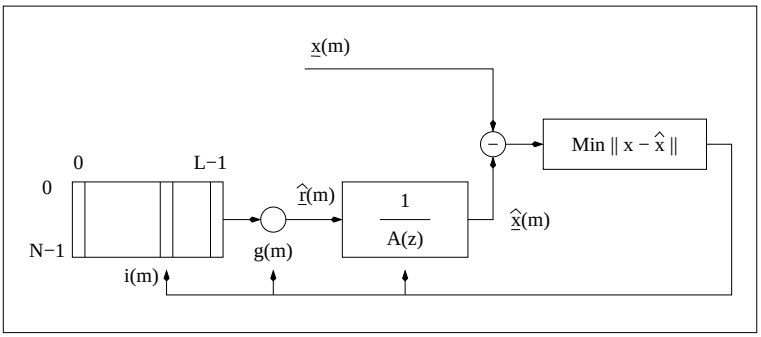

FIG. 6.3 – Schéma de principe du codeur CELP.

les N échantillons de la fenêtre d'analyse courante $\underline{x} = [x(0) \cdots x(N-1)]$, il s'agit de déterminer les coefficients $a_1 \cdots a_P$ d'un filtre autoregressif d'ordre P, le numéro i d'un vecteur dans un dictionnaire et un gain g minimisant la norme du vecteur d'erreur. La minimisation conjointe étant difficile, la minimisation se fait en deux temps. Dans une première étape, on suppose que le filtre est soumis à une entrée blanche. On cherche donc à modéliser le signal de parole par un processus aléatoire AR. On fait une "analyse LPC". Une fois les coefficients du filtre déterminés, il s'agit de caractériser l'entrée de ce filtre.

On peut fournir un autre schéma de principe strictement équivalent au précédent[3], visualisé figure 6.4, en privilégiant l'approche "théorie de l'information" plutôt que l'approche "modélisation paramétrique" et faire les commentaires suivants.

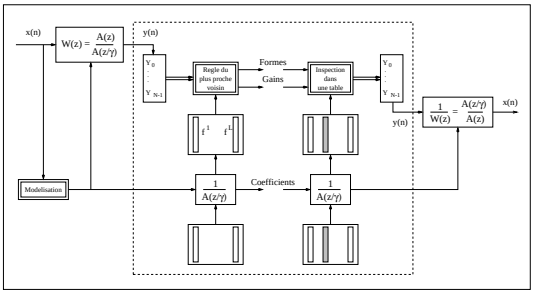

FIG. 6.4 – Autre schéma de principe du codeur CELP (avec fonction de pondération).

Il s'agit d'abord d'une quantification vectorielle dite "gain-forme". On a déjà vu dans le cadre du cours sur la quantification vectorielle que minimiser $||\underline{x} - \hat{g}^j \underline{\hat{x}}^k||$ relativement à j et k plutôt que $||\underline{x} - \underline{\hat{x}}^k||$ relativement à k uniquement permet de gérer commodément l'évolution au cours du temps de la puissance instantanée du signal. On joue sur le fait que la "forme" (un vecteur

[3] exceptée la fonction de pondération introduite plus tard

normé) qui caractérise le contenu spectral du signal et le gain (un scalaire) qui caractérise sa puissance sont deux informations assez décorrélées.

On réalise ensuite une adaptation régulière du dictionnaire de quantification vectorielle à la statistique locale du signal par une opération de filtrage. On exploite un modèle de production actualisé toutes les vingtaines de millisecondes, tous les $N = 160$ échantillons. On dispose de deux niveaux de dictionnaires. Le premier dictionnaire, composé des vecteurs $\{\underline{c}^0 \cdots \underline{c}^{L-1}\}$, est généralement choisi de façon à couvrir à peu près uniformément l'espace R^N. Il est déterminé une bonne fois pour toutes et il est connu en termes identiques à l'émetteur et au récepteur. Le second dictionnaire est réactualisé régulièrement par filtrage. Les paramètres du filtre sont calculés en réalisant une analyse LPC à partir des N échantillons du signal dans la fenêtre d'analyse courante. Cette opération de filtrage a pour but de partitionner finement l'espace R^N là où le vecteur $\underline{x}$ a de fortes chances d'apparaître et de façon grossière le reste de l'espace.

Deux autres caractéristiques seront introduites dans ce chapitre. La première est l'emploi d'une fonction de pondération dans le domaine fréquentiel, notée $W(z)$ sur le schéma, de façon à mettre en forme le bruit de quantification. Cette mise en forme est grossière contrairement au cas des codeurs de musique où on utilise un modèle d'audition élaboré comme on le verra plus tard car, pour les débits visés (de 4.8 à 16 kbit/s), le bruit de quantification est trop élevé et la bande utile (inférieure à 4 kHz) est trop réduite pour justifier un modèle précis. La seconde caractéristique (non représentée sur le schéma) est la nécessité d'introduire un "prédicteur à long terme" possédant une structure de type ADPCM.

6.3.2 Détermination des coefficients du filtre de synthèse

Le premier traitement consiste à déterminer puis coder les coefficients du filtre utilisé à la synthèse. Une modélisation par prédiction linéaire est quasi systématiquement utilisée. Le choix de l'ordre de prédiction P résulte d'un compromis. Il doit être suffisamment élevé pour reproduire correctement la structure formantique du signal de parole : un ordre 8 est nécessaire pour créer quatre pics dans le spectre et on sait que le signal de parole comporte généralement quatre formants. Inversement, l'ordre doit être le plus faible possible pour économiser le débit. On choisit donc habituellement P compris entre 8 et 16.

Les coefficients du filtre doivent être codés. Dans la pratique, on ne quantifie pas directement les coefficients $a_1 \cdots a_P$ du filtre $A(z)$ car ils ont de mauvaises propriétés de codage, en particulier on maîtrise mal leur dynamique. On préfère quantifier, de façon scalaire ou vectorielle, les coefficients $k_1 \cdots k_P$ du filtre en treillis correspondant. Ces coefficients peuvent être calculés à partir des coefficients de corrélation normalisés $\rho_1 \cdots \rho_P$ en utilisant l'algorithme de Levinson. Ils possèdent la bonne propriété d'être toujours compris entre -1 et $+1$ (si le filtre est stable) mais la distribution statistique de ces coefficients n'est pas du tout uniforme dans cet intervalle. Comme il est plus simple de quantifier un scalaire présentant une distribution relativement uniforme, on peut faire subir à ces coefficients une transformation non-linéaire de la forme

$$K_i = \log \frac{1 + k_i}{1 - k_i}.$$

On sait, en effet, que, lorsque les pôles de la fonction de transfert $1/A(z)$ se rapprochent du cercle unité ou, de façon équivalente, lorsque les k_i prennent des valeurs voisines de 1, la réponse en fréquence $1/A(f)$ présente un maximum de plus en plus accentué. On appelle ces nouveaux coefficients les *Log Area Ratios*.

Il existe une quatrième représentation de ces coefficients équivalente aux trois représentations précédentes possédant des propriétés de codage encore meilleures. Il s'agit des *Line Spectrum Pairs* [36]. On en donne une présentation très succincte. A partir du polynôme d'ordre P

$$A(z) = 1 + \sum_{i=1}^{P} a_i z^{-1}$$

on construit deux nouveaux polynômes d'ordre $P+1$

$$\begin{aligned} B_1(z) &= A(z) + z^{-P-1}A(z^{-1}) \\ B_2(z) &= A(z) - z^{-P-1}A(z^{-1}). \end{aligned}$$

On montre que ces deux polynômes ont les propriétés suivantes.

- Le polynôme $B_1(z)$ est un polynôme symétrique. Le polynôme $B_2(z)$ est un polynôme antisymétrique.
- Si toutes les racines de $A(z)$ sont à l'intérieur du cercle unité, toutes les racines de $B_1(z)$ et de $B_2(z)$ sont sur le cercle unité.
- Les racines de $B_1(z)$ et $B_2(z)$ apparaissent de façon alternée sur le cercle unité.
- Si P est pair, on peut écrire $B_1(z)$ et $B_2(z)$ sous la forme

$$\begin{aligned} B_1(z) &= (1+z^{-1}) \prod_{i=1}^{P/2} (1 - 2\cos\phi_{2i-1} z^{-1} + z^{-2}) \\ B_2(z) &= (1-z^{-1}) \prod_{i=1}^{P/2} (1 - 2\cos\phi_{2i} z^{-1} + z^{-2}). \end{aligned}$$

  Si P est impair, on obtient

$$\begin{aligned} B_1(z) &= \prod_{i=1}^{(P+1)/2} (1 - 2\cos\phi_{2i-1} z^{-1} + z^{-2}) \\ B_2(z) &= (1-z^{-2}) \prod_{i=1}^{(P-1)/2} (1 - 2\cos\phi_{2i} z^{-1} + z^{-2}). \end{aligned}$$

- Les Line Spectrum Pairs $\phi_1 \cdots \phi_P$ vérifient les relations

$$0 < \phi_1 < \cdots < \phi_P < \pi.$$

Connaissant les coefficients a_i, on en déduit les coefficients ϕ_i. Réciproquement, connaissant les coefficients ϕ_i, on en déduit les coefficients a_i puisque

$$A(z) = \frac{B_1(z) + B_2(z)}{2}.$$

La quantification des coefficients ϕ_i doit conserver les inégalités précédentes pour maintenir la stabilité du filtre de synthèse. Si l'on quantifie la différence entre deux coefficients successifs, ces relations sont automatiquement conservées.

On montre, figure 6.5, les racines du polynôme $A(z)$ (croix bleues), du polynôme $B_1(z)$ (ronds rouges) et du polynôme $B_2(z)$ (carrés noirs) lorsque l'analyse LPC a été faite à l'ordre $P = 12$ sur le son voisé représenté figure 6.2. On observe bien l'alternance des racines de $B_1(z)$ et $B_2(z)$ sur le cercle unité.

6.3.3 Modélisation de l'excitation

Introduction d'un facteur perceptuel

Les fonctions de coût quadratiques se prêtent bien aux calculs : elles possèdent la bonne propriété de fournir un système linéaire lorsque l'on dérive ce critère par rapport aux paramètres inconnus. Par contre, ce critère n'est pas forcément bien adapté à notre système auditif. Une correction perceptive est très largement utilisée pour pallier cet inconvénient [37]. On rajoute

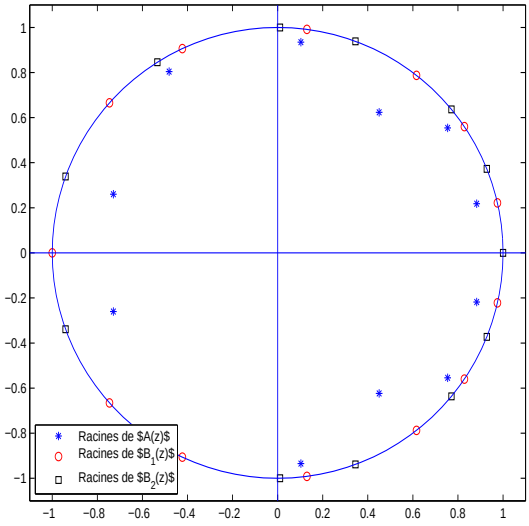

FIG. 6.5 – Coefficients LSP : racines des polynômes $A(z)$, $B_1(z)$ et $B_2(z)$.

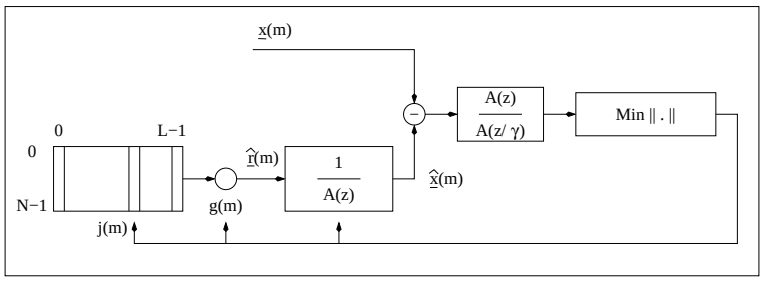

FIG. 6.6 – Introduction d'une fonction de pondération $W(z) = A(z)/A(z/\gamma)$.

une fonction de pondération, sous la forme d'un filtre de fonction de transfert W(z), avant le critère de minimisation comme l'indique la figure 6.6.

Tout un développement sur les phénomènes de masquage d'un son par un autre sera donné ultérieurement. Disons simplement maintenant que le bruit dû à la quantification est moins perceptible lorsque le signal a beaucoup d'énergie. On dit que le signal masque le bruit. Il n'est pas possible de jouer sur la puissance totale du bruit de quantification. Par contre il est possible de modifier la forme spectrale du bruit. On cherche donc une fonction de pondération qui attribue moins d'importance aux zones fréquentielles énergétiques c'est-à-dire aux zones formantiques. On montre que la fonction de transfert $W(z) = A(z)/A(z/\gamma)$ avec $0 < \gamma < 1$ joue ce rôle. En effet, si on note

$$A(z) = 1 + a_1 z^{-1} + \cdots + a_P z^{-P} = \prod_{i=1}^{P} (1 - p_i z^{-1})$$

où p_i spécifie la i^{eme} racine du polynôme $A(z)$, on remarque que

$$A(\frac{z}{\gamma}) = 1 + a_1 \gamma z^{-1} + \cdots + a_P \gamma^P z^{-P} = \prod_{i=1}^{P} (1 - \gamma p_i z^{-1}).$$

Le module de la réponse en fréquence du filtre $1/A(z/\gamma)$ présente des pics moins accentués que celui du filtre $1/A(z)$ puisque les pôles du filtre $1/A(z/\gamma)$ sont ramenés vers le centre du cercle unité par rapport à ceux du filtre $1/A(z)$. Le module de la réponse en fréquence du filtre

$W(z) = A(z)/A(z/\gamma)$ a donc la forme souhaitée comme le montre le tracé centré *grosso modo* à 0 dB de la figure 6.7.

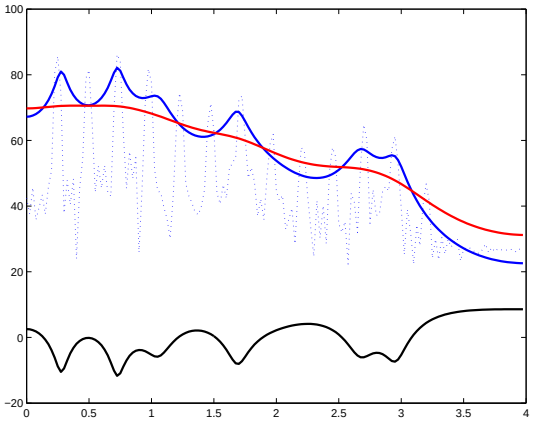

FIG. 6.7 – Réponses en fréquences des filtres $1/A(z)$ en bleu, $1/A(z/\gamma)$ en rouge et $A(z)/A(z/\gamma)$ en noir pour un son voisé dont le spectre est visualisé en pointillés. Dans cet exemple $\gamma = 0.8$.

Le diagramme donnant le principe de la modélisation devient celui de la figure 6.8. Ce dia-

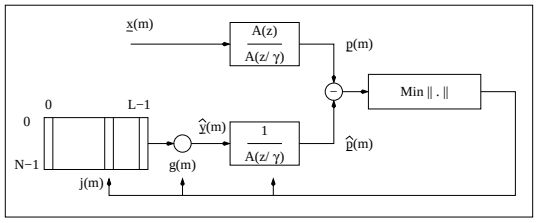

FIG. 6.8 – Modélisation du signal perceptuel.

gramme met clairement en évidence le fait que l'on cherche à modéliser le signal *perceptuel* $\underline{p}$ par $\underline{\hat{p}}$. On appelle par la suite filtre perceptuel, le filtre caractérisé par la fonction de transfert $1/A(z/\gamma)$. Le choix de la valeur numérique du facteur perceptuel γ permet de moduler la fonction de pondération à sa convenance. Pour $\gamma = 1$, tout se passe comme si on n'utilisait pas de fonction de pondération ; on effectue une modélisation du signal original. L'erreur de reconstruction $\underline{x} - \underline{\hat{x}}$ sera approximativement blanche. Pour $\gamma = 0$, on réalise une modélisation du signal résiduel. L'erreur de reconstruction aura la forme spectrale du signal original. On choisit généralement γ voisin de 0.8.

Choix du modèle d'excitation

Le schéma de la figure 6.3 laisse penser que l'on choisit, comme modèle pour l'excitation, un vecteur unique issu d'un dictionnaire et un seul gain. Pour plus de généralité, on prendra une entrée de la forme

$$\underline{\hat{y}} = \sum_{k=1}^{K} g_k \underline{c}^{j(k)}$$

où K est le nombre de vecteurs qui composent cette excitation, l'ordre de modélisation de l'excitation (habituellement $K = 2$ ou 3). *A priori*, chaque vecteur peut être issu d'un dictionnaire distinct comme on le verra par la suite mais on supposera pour commencer que ces K vecteurs sont issus du même dictionnaire $C = \{\underline{c}^0 \cdots \underline{c}^{L-1}\}$. Le problème consiste donc à chercher K

vecteurs $c^{j(1)} \cdots c^{j(K)}$ dans un dictionnaire d'excitation et K gains $g_1 \cdots g_K$ de façon à ce que le vecteur $\underline{\hat{y}}$ filtré par le filtre perceptuel $1/A(z/\gamma)$ donne le vecteur modélisé $\underline{\hat{p}}$ le plus ressemblant possible au vecteur $\underline{p}$.

Dictionnaire filtré

La forme du modèle de l'excitation étant fixée, il faut, maintenant, déterminer la valeur numérique des paramètres. On minimise $||\underline{p} - \underline{\hat{p}}||^2$ relativement aux indices $j(1) \cdots j(K)$ et aux gains $g_1 \cdots g_K$. Il faut donc mettre en évidence dans le critère ce qui est connu, c'est-à-dire ce qui a pu être calculé précédemment, et ce qui dépend explicitement des inconnues $j(k)$ et g_k. Le signal perceptuel modélisé a pour expression

$$\hat{p}(n) = \sum_{i=0}^{\infty} h(i)\hat{y}(n-i) \quad \text{pour } n = 0 \cdots N-1$$

où $h(n)$ est la réponse impulsionnelle du filtre perceptuel et où $n = 0$ caractérise par convention le premier échantillon de la fenêtre d'analyse courante. L'expression précédente se décompose en deux termes

$$\hat{p}(n) = \sum_{i=0}^{n} h(i)\hat{y}(n-i) + \sum_{i=n+1}^{\infty} h(i)\hat{y}(n-i).$$

Le premier terme est *a priori* inconnu mais le deuxième terme est connu puisqu'il fait intervenir $\hat{y}(n)$ pour $n < 0$ c'est à dire l'excitation du filtre de synthèse déterminée dans la fenêtre d'analyse précédente. Finalement le signal perceptuel modélisé s'écrit

$$\hat{p}(n) = \sum_{k=1}^{K} g_k \sum_{i=0}^{n} h(i)c^{j(k)}(n-i) + \sum_{i=n+1}^{\infty} h(i)\hat{y}(n-i).$$

On note

$$\underline{\hat{p}}^0 = [\sum_{i=1}^{\infty} h(i)\hat{y}(-i) \cdots \sum_{i=N}^{\infty} h(i)\hat{y}(N-1-i)]^t$$

la contribution dans la fenêtre courante de l'excitation provenant des fenêtres précédentes. On appelle ce vecteur le *ringing*, peut-être parce qu'une mauvaise gestion de ce terme entraîne un artefact à la fréquence f_e/N désagréable auditivement. On note

$$\underline{f}^j = [f^j(0) \cdots f^j(N-1)]^t$$

avec

$$f^j(n) = \sum_{i=0}^{n} h(i)c^j(n-i)$$

le résultat du filtrage du vecteur $\underline{c}^j$ par le filtre perceptuel partant de conditions initiales nulles. Cette opération de filtrage peut être caractérisée par l'expression matricielle

$$\underline{f}^j = \begin{bmatrix} h(0) & 0 & \cdots & 0 \\ h(1) & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ h(N-1) & \cdots & h(1) & h(0) \end{bmatrix} \underline{c}^j = H\underline{c}^j.$$

Le nouveau schéma de principe devient celui de la figure 6.9. On met en évidence le dictionnaire d'excitation C et le dictionnaire filtré F, tous les deux composés des L vecteurs notés respectivement $\underline{c}^j$ et $\underline{f}^j$.

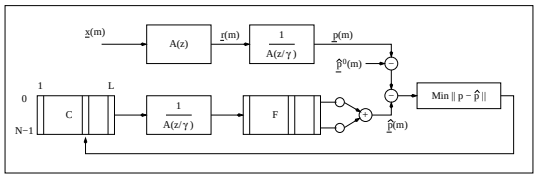

FIG. 6.9 – Modélisation du signal perceptuel par K vecteurs issus du dictionnaire filtré.

On peut faire la remarque suivante. Dans toutes les expressions précédentes, on a supposé, implicitement, que les filtres mis en jeu étaient invariants. En fait, ils sont simplement localement invariants. Le vecteur $\underline{\hat{p}}^0$ est donc la réponse libre du filtre dont les coefficients sont caractéristiques de la fenêtre courante. On ne touche pas au vecteur d'état du filtre lorsque l'on passe d'une fenêtre à l'autre.

Minimisation au sens des moindres carrés

A partir de maintenant, pour simplifier les notations, on appelle $\underline{p}$ le vecteur perceptuel auquel on a enlevé la contribution provenant des fenêtres précédentes. Le problème de la détermination de l'excitation dans un codeur CELP peut être exprimé de la façon suivante. Connaissant $\underline{p}$ et $\underline{f}^0 \cdots \underline{f}^{L-1}$, trouver les indices $j(1) \cdots j(K)$ et les gains $g_1 \cdots g_K$ de façon à minimiser

$$D = ||\underline{p} - \underline{\hat{p}}||^2 = ||\underline{p} - \sum_{k=1}^{K} g_k \underline{f}^{j(k)}||^2.$$

En notations matricielles, ce problème devient : étant donné une matrice F appartenant à R^{N*L}, composée des L vecteurs colonnes $\underline{f}^j$, et un vecteur $\underline{p}$ appartenant à R^N, extraire de F une matrice A appartenant à R^{N*K}, composée de K vecteurs colonnes parmi L, et trouver un vecteur $\underline{g}$ de dimension K de façon à minimiser

$$D = ||\underline{p} - A\underline{g}||^2 = ||\underline{p} - \underline{\hat{p}}||^2.$$

Il s'agit d'un problème classique de minimisation au sens des moindres carrés si l'on suppose connus les indices $j(1) \cdots j(K)$, c'est-à-dire la matrice A [38]. On détermine $\underline{g}$ à partir du système sur-dimensionné $\underline{p} = A\underline{g}$. On écrit que la meilleure approximation $\underline{\hat{p}}$ de $\underline{p}$ est la projection orthogonale de $\underline{p}$ dans le sous-espace engendré par les vecteurs colonnes de A ou que le vecteur $\underline{p} - \underline{\hat{p}}$ est orthogonal à tous les vecteurs engendrant ce sous-espace. On obtient les *équations normales*

$$(\underline{f}^{j(k)})^t(\underline{p} - A\underline{g}) = 0 \quad \text{pour} \quad k = 1 \cdots K$$

ou de façon matricielle

$$A^t A\underline{g} = A^t\underline{p}.$$

La matrice A est généralement de rang complet. La matrice A^tA est donc définie positive. Cette résolution se fait alors en utilisant la décomposition de Choleski, par exemple, ou par des algorithmes rapides si on impose une forme Toeplitz ou proche de Toeplitz à cette matrice. Malheureusement les indices sont inconnus et doivent être déterminés simultanément avec les gains.

Il existe un algorithme optimal : il suffit de déterminer toutes les matrices A correspondant à toutes les combinaisons de K vecteurs parmi L, de résoudre pour chaque matrice A les équations normales, d'évaluer le critère et de sélectionner la combinaison qui minimise ce critère. Sachant qu'il existe $L!/K!(L-K)!$ combinaisons possibles et que des ordres de grandeur pour K, L et

N sont $K = 3$, $L = 256$ et $N = 40$, on est amené à résoudre $256 * 255 * 254/6$ c'est-à-dire près de 3 millions de systèmes linéaires d'ordre 3 toutes les 5 ms ! Il est donc nécessaire de se limiter à un algorithme sous-optimal.

Précisons le problème. La minimisation de D consiste à choisir une combinaison $j(1) \cdots j(K)$, résoudre les équations normales $A^t A\underline{g} = A^t\underline{p}$, enfin sélectionner la combinaison qui maximise $\underline{g}^t A^t A\underline{g}$ puisque la minimisation de $||\underline{p} - \underline{\hat{p}}||^2$ est équivalente à la maximisation de $||\underline{\hat{p}}||^2$. En effet, il suffit d'appliquer le théorème de Pythagore

$$||\underline{p} - \underline{\hat{p}}||^2 = ||\underline{p}||^2 - ||\underline{\hat{p}}||^2$$

et de remarquer que $||\underline{p}||^2$ est constant. Puisque

$$\underline{g} = (A^t A)^{-1} A^t \underline{p}$$

le problème consiste finalement à sélectionner la combinaison $j(1) \cdots j(K)$ qui maximise

$$||\underline{\hat{p}}||^2 = \underline{p}^t A (A^t A)^{-1} A^t \underline{p}$$

puis à calculer les gains en résolvant, une seule fois, les équations normales correspondant à la combinaison choisie.

Algorithme itératif standard

La difficulté provient essentiellement de l'inversion de la matrice $A^t A$. Il existe deux façons (au moins) de simplifier le traitement. On se limite à la recherche d'un vecteur à la fois, l'expression $A^t A$ est alors un scalaire, le calcul des indices et des gains se fait de façon itérative. Ou alors, on impose des propriétés particulières à la matrice A, par exemple : $A^t A = I$ quelle que soit la combinaison. La première méthode est tout à fait standard. Elle a été proposée par [34] pour une excitation multi-impulsionnelle. Elle est largement utilisée dans la pratique mais elle est, malheureusement, sous optimale. La seconde méthode conduit à un algorithme optimal mais elle impose des contraintes trop fortes à la matrice F : elle doit être composée de vecteurs orthogonaux.

On présente, ici, l'algorithme itératif standard. On note $< \underline{x}, \underline{y} >$ le produit scalaire des deux vecteurs $\underline{x}$ et $\underline{y}$. A la première itération, un seul vecteur $\underline{f}^j$ est sélectionné. On a

$$A^t A =< \underline{f}^j, \underline{f}^j >$$

$$A^t \underline{p} =< \underline{f}^j, \underline{p} > .$$

On doit donc choisir l'index j qui maximise

$$< \underline{p}, \underline{f}^j >< \underline{f}^j, \underline{f}^j >^{-1} < \underline{f}^j, \underline{p} >= \frac{< \underline{f}^j, \underline{p} >^2}{< \underline{f}^j, \underline{f}^j >}$$

puis calculer le gain

$$g_1 = \frac{< \underline{f}^{j(1)}, \underline{p} >}{< \underline{f}^{j(1)}, \underline{f}^{j(1)} >}.$$

A la k^{eme} itération, la contribution des $k-1$ premiers vecteurs $\underline{f}^{j(i)}$ est retirée de $\underline{p}$

$$\underline{p}^k = \underline{p} - \sum_{i=1}^{k-1} g_i \underline{f}^{j(i)}$$

et un nouvel index $j(k)$ et un nouveau gain g_k sont calculés vérifiant

$$j(k) = \arg\max_j \frac{< \underline{f}^j, \underline{p}^k >^2}{< \underline{f}^j, \underline{f}^j >}$$

$$g_k = \frac{< \underline{f}^{j(k)}, \underline{p}^k >}{< \underline{f}^{j(k)}, \underline{f}^{j(k)} >}.$$

Choix du dictionnaire d'excitation

Cet algorithme est applicable quel que soit le contenu du dictionnaire d'excitation. Si on choisit $C = I$ où C est la matrice composée des vecteurs colonnes $\underline{c}^j$ et I la matrice identité de dimension $N * N$, on obtient une excitation "multi-impulsionnelle". Les indices sélectionnés $j(k)$ caractérisent les positions des impulsions choisies et les gains g_k définissent les amplitudes. Dans le codeur CELP classique, on initialise le dictionnaire d'excitation par des tirages d'une variable aléatoire gaussienne centrée ou on le construit par apprentissage en utilisant une variante de l'algorithme de Lloyd-Max. Dans ces deux cas, on peut vérifier que la complexité du traitement est assez importante. Pour réduire cette complexité, on peut chercher à définir un dictionnaire comportant une forte structure par exemple imposer des valeurs ternaires {+1, -1, 0} et choisir des positions régulières dans le dictionnaire pour les valeurs non nulles. C'est la particularité du codeur G.729 [39] qui est appelé pour cette raison le codeur ACELP (Algebraic CELP).

Introduction d'un dictionnaire adaptatif

On sait que, pour déterminer les coefficients du filtre $A(z)$, on minimise l'énergie de l'erreur de prédiction

$$D_1 = \sum_n [x(n) - \sum_{i=1}^{P} a_i x(n-i)]^2$$

par rapport aux P paramètres inconnus a_i. Il s'agit d'une prédiction dite à court terme puisque, pour prédire la valeur du signal à l'indice n, on utilise les P échantillons précédents. Une fois le calcul réalisé par simple résolution du système linéaire obtenu en dérivant D_1 par rapport aux P paramètres inconnus, on filtre le signal $x(n)$ par le filtre de fonction de transfert $A(z)$ d'ordre P. On obtient le signal résiduel à court terme $y(n)$. La visualisation de ce signal, spécialement pour des sons voisés, montre que toute la redondance placée dans le signal de parole n'a pas été extraite. Il reste une certaine périodicité comme le montre les tracés de la figure 6.10. Cette périodicité correspond, physiologiquement, à la période de vibration des cordes vocales. On peut chercher à caractériser cette information en introduisant deux nouveaux paramètres b et Q puis en minimisant l'énergie d'une nouvelle erreur de prédiction

$$D_2 = \sum_n [y(n) - by(n-Q)]^2$$

par rapport à ces deux paramètres inconnus. On parle alors de prédiction à long terme. On remarquera que cette minimisation ne peut pas être réalisée comme la précédente puisque, pour D_1, P est fixé, alors que pour D_2, Q est un paramètre à déterminer.

Minimisation en "boucle ouverte"

Pour calculer les valeurs optimales de b et Q, il suffit de dériver D_2 par rapport à b pour obtenir b optimal comme une fonction de Q, de reporter cette valeur dans D_2 puis de choisir la valeur de Q qui minimise le critère. Ce n'est pas forcément la solution la plus adaptée. On préfère, généralement, une solution en boucle fermée.

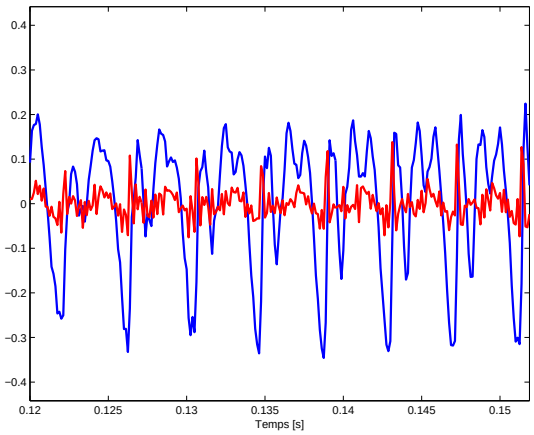

FIG. 6.10 – Signal de parole $x(n)$ et signal résiduel $y(n)$ pour un locuteur féminin.

Minimisation en "boucle fermée"

Introduisons la fonction de transfert

$$B(z) = 1 - bz^{-Q}.$$

A la synthèse, on utilise le filtre inverse $1/B(z)$ qui doit être placé en amont du filtre de fonction de transfert $1/A(z)$. Le diagramme fonctionnel correspondant est donné figure 6.11. Reprenons

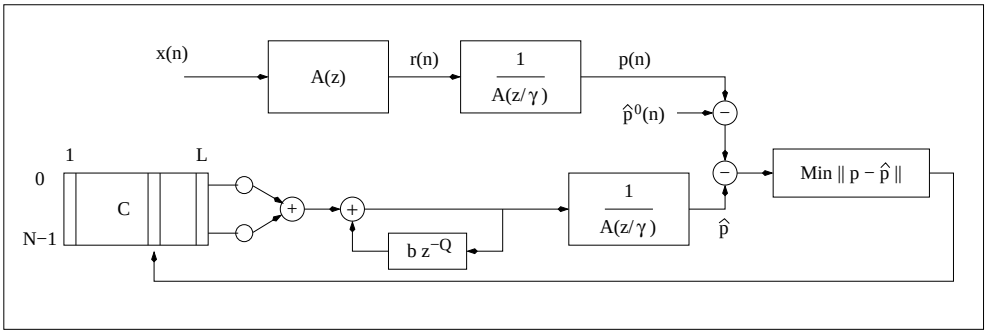

FIG. 6.11 – Prédiction à court terme et à long terme.

le développement présenté précédemment en supposant b et Q prédéterminés. On cherche des vecteurs $\underline{c}^{j(k)}$ dans le dictionnaire d'excitation et des gains g_k de façon à ce que le vecteur $\sum_{k=1}^{K} g_k \underline{c}^{j(k)}$ filtré par le filtre $1/B(z)$ puis par le filtre perceptuel $1/A(z/\gamma)$ donne le vecteur modélisé $\underline{\hat{p}}$ le plus ressemblant possible au vecteur $\underline{p}$. On a vu que le signal perceptuel modélisé avait pour expression

$$\hat{p}(n) = \sum_{i=0}^{n} h(i)\hat{y}(n-i) + \sum_{i=n+1}^{\infty} h(i)\hat{y}(n-i) \quad \text{pour } n = 0 \cdots N-1.$$

Mais $\hat{y}(n)$ pour $n \geq 0$, *a priori* inconnu, se décompose aussi en une partie inconnue qui ne dépend que des $\underline{c}^{j(k)}$ et g_k et une partie connue

$$\hat{y}(n) = \sum_{k=1}^{K} g_k c^{j(k)}(n) + b\hat{y}(n-Q)$$

si on admet les hypothèses que les paramètres b et Q du prédicteur à long terme ont été déterminés et que

$$n - Q < 0 \quad \forall n \in 0 \cdots N-1$$

c'est-à-dire

$$Q \geq N.$$

La valeur du décalage doit donc être supérieure ou égale à la taille de la fenêtre d'analyse. Finalement le signal perceptuel modélisé s'écrit

$$\hat{p}(n) = \sum_{k=1}^{K} g_k \sum_{i=0}^{n} h(i) c^{j(k)}(n-i) + b \sum_{i=0}^{n} h(i)\hat{y}(n-i-Q) + \sum_{i=n+1}^{\infty} h(i)\hat{y}(n-i).$$

On appelle

$$\underline{\hat{p}}^k = [\hat{p}^k(0) \cdots \hat{p}^k(N-1)]^t$$

avec

$$\begin{aligned}
\hat{p}^0(n) &= \sum_{i=n+1}^{\infty} h(i)\hat{y}(n-i) \\
\hat{p}^1(n) &= b \sum_{i=0}^{n} h(i)\hat{y}(n-i-Q) \\
\hat{p}^k(n) &= g_k \sum_{i=0}^{n} h(i) c^{j(k)}(n-i).
\end{aligned}$$

On remarque que $\underline{\hat{p}}^1$ et $\underline{\hat{p}}^k$ ont exactement la même forme et s'interprètent de la même façon, comme le filtrage d'un vecteur connu par le filtre perceptuel partant de conditions initiales nulles et d'une pondération par un gain. L'algorithme standard consiste donc d'abord à minimiser le critère

$$||(\underline{p} - \underline{\hat{p}}^0) - \underline{\hat{p}}^1||^2$$

pour déterminer Q et b puis à minimiser le critère

$$||(\underline{p} - \underline{\hat{p}}^0 - \underline{\hat{p}}^1) - \underline{\hat{p}}^2||^2$$

pour déterminer j(2) et g_2, etc. On remarque que les deux paramètres du prédicteur à long terme Q et b peuvent être calculés exactement de la même manière que les paramètres $j(k)$ et g_k à la condition de construire un dictionnaire *adaptatif* comportant l'excitation passée

$$C = \begin{bmatrix}
\hat{y}(-Q_{max}) & \cdots & \hat{y}(-2N) & \hat{y}(-2N+1) & \cdots & \hat{y}(-N) \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\hat{y}(-Q_{max}+N-1) & \cdots & \hat{y}(-N-1) & \hat{y}(-N) & \cdots & \hat{y}(-1)
\end{bmatrix}.$$

Ce dictionnaire possède deux propriétés intéressantes. D'abord, la matrice correspondante a une structure de Toeplitz. On pourrait voir que cette propriété permet de réduire le nombre d'opérations mises en jeu lorsque l'on filtre ce dictionnaire. Ensuite, lorsque l'on passe d'une fenêtre d'analyse à la suivante, l'ensemble du dictionnaire n'est pas remis en cause. Uniquement N vecteurs doivent être actualisés. Les autres s'en déduisent par translation vers la gauche.

Remarques

La contrainte $Q \geq N$ est trop forte dans la pratique. Il est nécessaire d'introduire un traitement par sous-fenêtre et de déterminer un modèle pour l'excitation pour chaque sous-fenêtre. En effet la fréquence fondamentale moyenne est de l'ordre de 100 Hz pour un locuteur masculin et de 250 Hz pour un locuteur féminin. Cela veut dire que les valeurs probables de Q sont respectivement égales à 80 et 32. Comme la valeur habituellement choisie pour N est de 160, il semble nécessaire de diviser la fenêtre d'analyse en, au moins, 5 sous-fenêtres mais, généralement on se limite à 4 sous-fenêtres de $N' = 40$ échantillons car cela deviendrait trop coûteux en terme de débit.

On peut prolonger le dictionnaire C vers la droite en le construisant de la façon suivante. La première fois, on utilise les $N-1$ échantillons disponibles $\hat{y}(-N+1)\cdots\hat{y}(-1)$ et on les complète par $\hat{y}(-N+1)$, la deuxième fois on utilise les $N-2$ échantillons $\hat{y}(-N+2)\cdots\hat{y}(-1)$ et on les complète par $\hat{y}(-N+2)\hat{y}(-N+3)$, etc. On s'arrête à l'indice Q_{min}. Ceci résout artificiellement le problème de la taille trop élevée de la sous-fenêtre pour la période fondamentale d'un locuteur féminin. Il suffit de choisir $Q_{min} = 20$, par exemple. On prend

$$Q_{max} = Q_{min} + 127$$

si l'on décide de coder l'indice sur 7 bits.

6.3.4 Conclusion

Nous ne donnons ici que les principes sur lesquels sont bâtis les codeurs de parole de type CELP. Pour obtenir du signal reconstruit de qualité convenable, il est nécessaire de régler tous les paramètres mis en jeu de façon précise et de rajouter toute une série "d'astuces". Pour en avoir une certaine idée, on pourra consulter le papier [40].

On donne simplement ici les informations transmises dans le réseau dans le cas du codeur G.729 à 8 kbit/s. Les $P = 10$ coefficients du filtre de synthèse $1/A(z)$ sont actualisés toutes les 10 ms (fenêtre d'analyse de 80 échantillons). Le codage des "Line Spectrum Pairs" est réalisé sur 18 bits. L'entrée du filtre de synthèse est la somme de deux signaux actualisés toutes les 5 ms. Le premier correspond au prédicteur à long terme. Le paramètre Q et le gain sont codés respectivement sur 7 bits et 3 bits. Le second est issu d'un dictionnaire de quantification vectorielle. Le gain est codé sur 4 bits et l'indice sur 13+4 bits conformément à ce qu'indique la table 6.1. Toutes les 10 ms il faut donc transmettre $18 + 2 \times (3 + 7 + 4 + 17) = 80$ bits ce qui entraîne bien un débit de 8 kbit/s.

Impulsions	Amplitude	Positions	Bits
0	±1	0, 5, 10, 15, 20, 25, 30, 35	1 + 3
1	±1	1, 6, 11, 16, 21, 26, 31, 36	1 + 3
2	±1	2, 7, 12, 17, 22, 27, 32, 37	1 + 3
3	±1	3, 8, 13, 18, 23, 28, 33, 38 4, 9, 14, 19, 24, 29, 34, 39	1 + 4

TAB. 6.1 – Tous les 40 échantillons le codeur G.729 sélectionne 4 "impulsions" dont on indique les positions possibles.

On peut aussi fournir quelques informations concernant le codeur 3GPP AMR-WB (Adaptive Multi-Rate Wideband UIT-T G.722.2) [25] qui est le premier codeur adapté pour les réseaux fixes ou mobiles. Il supprime la nécessité de transcodages. Comme son nom l'indique plusieurs débits sont possibles. Ils varient entre 6.6 et 23.85 kbit/s. Le fait qu'il soit en bande élargie améliore sensiblement la qualité. L'introduction des basses fréquences 50-200 Hz rend la voix plus naturelle

et améliore l'effet de présence. L'extension 3.4-7 kHz entraîne une plus grande intelligibilité. C'est un codeur de type ACELP très comparable au G.729 mais il a subit une modification du filtrage perceptuel (extension à la bande élargie) et une modification de l'exploitation de l'information de pitch puisque la parole n'a pas de structure harmonique sur toute la bande 50 Hz - 7 kHz. Sa principale originalité est l'introduction d'un très grand dictionnaire d'excitation ($\log_2 L = 88$ bits).

Chapitre 7

Codeurs de signaux de musique

7.1 Principe des codeurs "perceptuels"

On montre, figure 7.1, l'amplitude d'un signal de violon en fonction du temps (à gauche) et la répartition de la puissance de ce signal en fonction de la fréquence (à droite). Comme pour

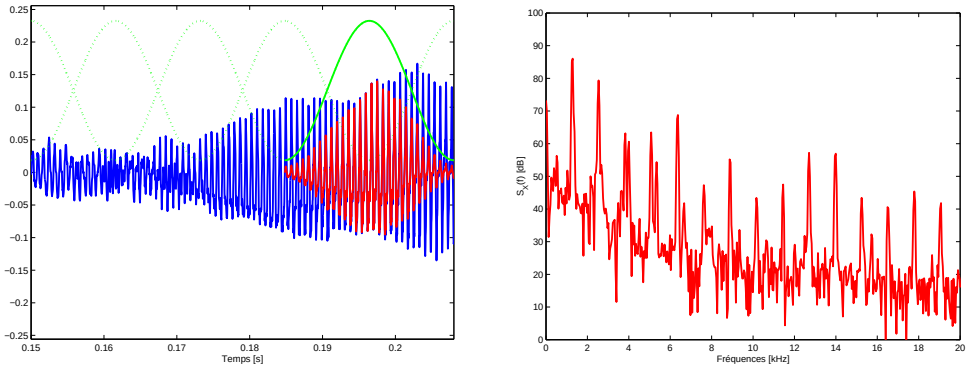

FIG. 7.1 – Signal de violon dans le domaine temporel (à gauche) et dans le domaine fréquentiel (à droite)

les codeurs de parole, un algorithme de compression est essentiellement une boucle consistant à remplir d'abord un buffer avec N échantillons comme le montre le schéma de la figure 7.2, à traiter ensuite ces N échantillons puis à passer à la fenêtre d'analyse suivante. La fenêtre d'analyse est toujours avec recouvrement, le décalage entre deux fenêtres d'analyse étant caractérisé par le paramètre $M < N$. Le vecteur $\underline{x}(m)$ est donc de la forme

$$\underline{x}(m) = [x(mM), \cdots, x(mM + N - 1)]^t \otimes [v(0), \cdots, v(N - 1)]^t$$

où l'opérateur $\otimes$ correspond à une multiplication de deux vecteurs composante par composante et où $\underline{v}$ est une fenêtre de pondération. Les fenêtres d'analyse étant généralement d'une vingtaine de ms, le paramètre N qui devrait être égal à $44.1 \times 20 = 882$ est égal à 512 (MPEG-1 et AC3) ou 2048 (MPEG-2 AAC). La valeur du paramètre M dépend du codeur : $M = 32 \ll N/2$ pour le codeur MPEG-1, $M = N/2$ pour les codeurs AC3 et MPEG-2 AAC.

Le schéma de principe d'un codeur perceptuel montre que celui-ci se décompose en trois modules distincts qui sont activés chaque fenêtre d'analyse : une transformation temps-fréquences de la forme $\underline{X}(m) = H\underline{x}(m)$, une allocation de bits sous le contrôle d'un modèle d'audition et une quantification scalaire ou vectorielle des composantes du vecteur $\underline{X}(m)$ suivie d'un codage entropique. La chaîne binaire transmise dans le canal est constituée des mots de code issus du

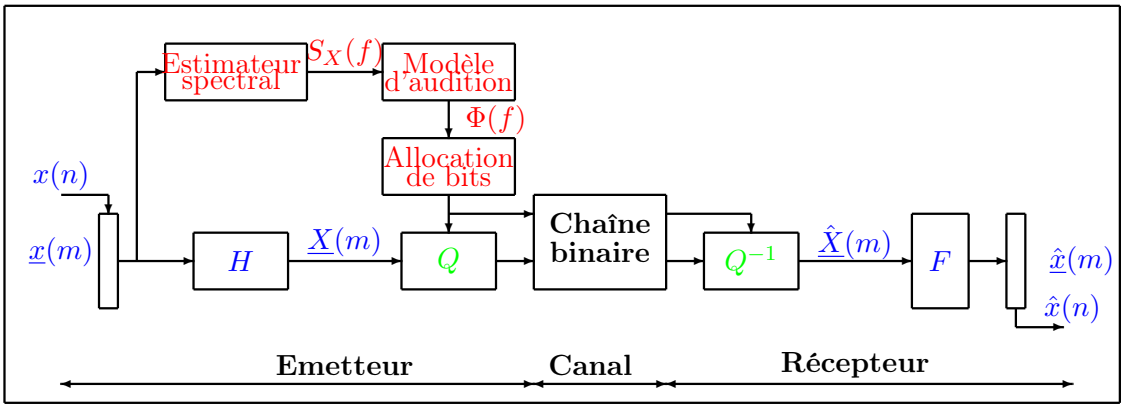

FIG. 7.2 – Schéma de principe d'un codeur perceptuel

codage entropique plus des informations adjacentes comme par exemple le résultat de l'allocation de bits, information *a priori* nécessaire au récepteur. Au récepteur, il faut construire le vecteur $\hat{\underline{X}}(m)$, revenir dans le domaine temporel par la transformée "inverse" $\hat{\underline{x}}(m) = F\hat{\underline{X}}(m)$ puis extraire à la cadence f_e les échantillons $\hat{x}(n)$. Dans le codeur MPEG-2 AAC, le calcul de l'erreur de reconstruction $q(n) = x(n) - \hat{x}(n)$ est explicite à l'émetteur en disposant d'une copie locale du récepteur. Cette information est exploitée dans le module d'allocation de bits. On parle de codage "en boucle fermée" comme dans le cas des codeurs de parole.

On rappelle que l'on a montré, section 3.2, l'équivalence entre bancs de filtres et transformées. Les modules des composantes du vecteur $\underline{X}(m)$ ont une interprétation fréquentielle si on choisit correctement la transformée, la MDCT en général. On admettra, dans cette section introductive, que ces modules fournissent une bonne estimation de la densité spectrale de puissance $S_X(f)$ du signal de musique dans la fenêtre d'analyse courante.

Les codeurs perceptuels cherchent à éliminer les composantes du vecteur $\underline{X}(m)$ qui ne sont pas audibles et à quantifier avec le minimum de bits les composantes restantes. Il faut définir un critère d'inaudibilité : c'est le rôle du modèle d'audition. Les résultats de psychoacoustique, développés à la section 7.5, montre qu'un son pur (une sinusoïde) ou un bruit à bande étroite peut être inaudible (masqué) en présence d'un autre son pur ou bruit à bande étroite. Sur le tracé de gauche de la figure 7.3, on a isolé une composante de la DSP $S_X(f)$. La courbe ayant

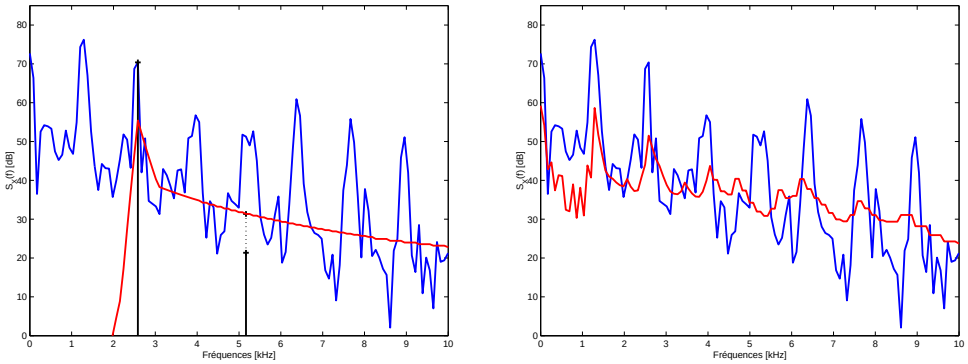

FIG. 7.3 – Courbe de masquage (à gauche). Seuil de masquage (à droite). On ne montre que leurs contributions dans la bande $[0 - 10]$ kHz

vaguement la forme d'un chapeau donne la puissance de la sinusoïde masquée juste à la limite de l'inaudibilité. Comme, en réalité, les N échantillons d'un son de musique dans la fenêtre

d'analyse courante s'expriment sous la forme d'une somme de M composantes fréquentielles, il faut généraliser ces résultats de psychoacoustique et sommer les différentes contributions. On obtient le "seuil de masquage" $\Phi(f)$ visualisé sur le tracé de droite en superposition avec la DSP $S_X(f)$. L'erreur due à l'opération de quantification entraîne une perturbation inaudible si sa DSP $S_Q(f)$ vérifie la relation

$$S_Q(f) < \Phi(f)$$

quel que soit f et quelle que soit la fenêtre d'analyse. La formule (4.17) dont on rappelle l'expression

$$b \geq \frac{1}{2}\int_{-1/2}^{+1/2} \max[0, \log_2 \frac{S_X(f)}{S_Q(f)}]\ df$$

donnant le débit nécessaire et suffisant pour coder une source de DSP $S_X(f)$ avec une distortion de DSP $S_Q(f)$, fournit la marche à suivre. Comme $[S_Q(f)]_{max} = \Phi(f)$, il faut partitionner finement l'axe des fréquences, déterminer les bandes de fréquence vérifiant $S_X(f) > \Phi(f)$ puis allouer les bits en fonction du rapport $S_X(f)/\Phi(f)$ en appliquant la règle des 6 dB par bit. Le tracé de la figure 7.4 montre que, dans cette fenêtre d'analyse, approximativement 50 % des

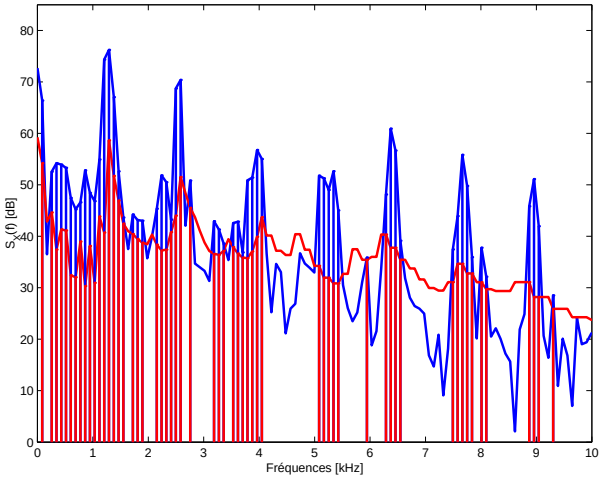

FIG. 7.4 – Elimination des composantes masquées.

composantes peuvent être éliminées et que le rapport "signal sur masque" moyen est de l'ordre de 12 dB ce qui réclame donc 2 bits en moyenne par composante. Le taux de compression est donc de l'ordre de $2 \times 16/2 = 16$. Pour du signal de musique au format CD, le débit le plus faible tout en respectant la contrainte d'inaudibilité (dans cette fenêtre) est de l'ordre d'une quarantaine de kbit/s. On voit tout de suite que tout dépend de la précision avec laquelle on a défini le modèle psychoacoustique. Définir un "bon" modèle est un problème très délicat.

En réalité, la définition d'un codeur audio ne se résume pas à cette démarche. Toute une série de problèmes spécifiques doivent être étudiés. Les deux prochaines sections décrivant les deux codeurs perceptuels de base devraient montrer que la réalisation d'un codeur audio présentant de bonnes performances est le résultat de nombreux compromis assez subtils. Si on désire une étude beaucoup plus élaborée, on se référera à [41].

7.2 Codeur MPEG-1 Layer1

On ne donne ici que quelques informations succinctes relatives à la partie audio de la norme internationale ISO/CEI 11172 [27]. Cette norme autorise 3 fréquences d'échantillonnage différentes : 32, 44.1 et 48 kHz. Elle autorise également une large gamme de débit. Dans cette

section, on supposera la fréquence d'échantillonnage égale à 44.1 kHz et les débits visés compris entre 64 et 96 kbit/s par voie. Toutes les tables fournies seront donc uniquement celles relatives à cette fréquence d'échantillonnage et à ces débits.

7.2.1 Transformation temps/fréquences

La transformation temps/fréquences utilisée est un banc de $M = 32$ filtres pseudo-QMF[1] réalisant une partition uniforme de l'axe des fréquences comme le montre le tracé de la figure 7.5 qui ne visualise que les réponses en fréquence des premiers filtres du banc. Ce banc de filtres n'est

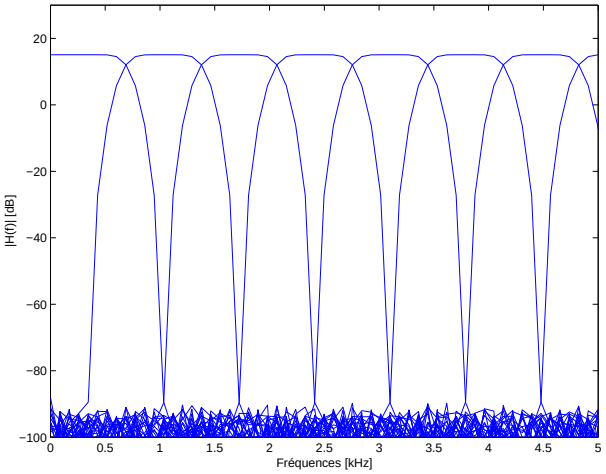

FIG. 7.5 – Réponses en fréquence du banc de filtres PQMF dans la bande [0 - 5] kHz.

pas à reconstruction parfaite mais, en absence de quantification, le RSB est supérieur à 90 dB ce qui suffisant puisque le RSB propre au format CD est lui même de cet ordre de grandeur. Ce banc de filtres délivre M signaux de sous-bande qui sont sous-échantillonnés par un facteur 32. On obtient les échantillons de sous-bande[2] $y_k(m)$. Indépendamment dans chaque sous-bande, le codeur construit un vecteur $\underline{y}_k$ regroupant 12 échantillons tous les 32*12=384 échantillons $x(n)$ correspondant approximativement à une dizaine de ms. Pour chaque vecteur $\underline{y}_k$, la composante la plus importante en valeur absolue parmi les 12 permet de définir un *facteur d'échelle* g_k et d'en déduire un vecteur normalisé $\underline{a}_k$. Chaque facteur d'échelle est exprimé en dB puis quantifié à l'aide d'un dictionnaire couvrant plus de 100 dB par pas de 2 dB ce qui nécessite 6 bits.

Le choix de 12 échantillons est le résultat d'un compromis. Pour des valeurs plus faibles, le débit associé aux facteurs d'échelle devient trop important. Pour des valeurs plus élevées, des phénomènes d'écho commencent à être audibles car il n'y a plus de masquage temporel. *A priori*, ce codeur présente une bonne résolution temporelle (les signaux de sous-bande sont actualisés fréquemment) mais une mauvaise résolution fréquentielle (les largeurs des bandes passantes des filtres du banc sont de l'ordre de $22/32 = 0.7$ kHz).

7.2.2 Modélisation psychoacoustique et allocation de bits

Les échantillons du signal $x(n)$ ayant contribué à la détermination des M vecteurs $\underline{a}_k$ sont aussi utilisés pour réaliser une estimation de la densité spectrale de puissance $\hat{S}_X(f)$ de ce signal dans la fenêtre d'analyse courante. A partir de $\hat{S}_X(f)$, le codeur calcule un seuil de masquage

[1] Filtres "miroir en quadrature"

[2] Pour mettre clairement en évidence le fait que ce sont d'abord des signaux avant d'être des composantes d'un vecteur ayant une interprétation fréquentielle, on a gardé la notation utilisée à la section 3.2 plutôt que la notation $\underline{X}(m)$ de ce chapitre.

$\Phi(f)$ en utilisant un modèle psycho-acoustique puis un *rapport signal sur masque* $\hat{S}_X(f)/\Phi(f)$. A partir de ces M rapports signal sur masque, le codeur réalise une allocation de bits c'est à dire détermine le nombre de bits b_k avec lesquels sera quantifiée chaque composante du vecteur $\underline{a}_k$. L'algorithme adopté est l'algorithme "greedy" standard.

7.2.3 Quantification

La quantification de chacune des composantes des vecteurs normalisés $\underline{a}_k$ est ensuite réalisée en utilisant des quantificateurs scalaires uniformes dans l'intervalle [-1, +1] avec un pas de quantification fonction de b_k.

Chaque signal de sous-bande dispose d'un certain nombre de quantificateurs possibles. Chaque quantificateur est caractérisé par un nombre de pas de quantification L et un rapport signal sur bruit σ_X^2/σ_Q^2 (un point dans un plan "débit-distorsion"). Les valeurs adoptées par la norme ISO pour les rapports signal sur bruit en fonction du nombre de pas de quantification sont données table 7.1. On donne également, dans un but de comparaison, le rapport signal sur bruit théorique

L	SNR_{ISO}	$20\log_{10} L$	b
3	7.00	9.54	1.67
5	11.00	13.98	2.33
7	16.00	16.90	3
9	20.84	19.08	3.33
15	25.28	23.52	4
31	31.59	29.82	5
63	37.75	35.99	6
127	43.84	42.08	7
255	49.89	48.13	8
511	55.93	54.17	9
1023	61.96	60.20	10
2047	67.98	66.22	11
4095	74.01	72.25	12
8191	80.03	78.27	13
16383	86.05	84.29	14
32767	92.01	90.31	15
65535	98.01	96.33	16

TAB. 7.1 – Rapports signal sur bruit en fonction du nombre de pas de quantification.

pour une source uniforme. Il est donné par la formule (1.5) en posant $c(1) = 1$.

On remarque que le nombre de pas de quantification est toujours un nombre impair de façon à ce que la valeur 0 soit toujours un représentant possible[3]. Le problème qui en résulte est que, pour des valeurs faibles de L, il existe une différence significative entre $b = \log_2 L$ et la valeur entière immédiatement supérieure $\lceil \log_2 L \rceil$. Il y a donc un risque de gaspillage de bits. Dans la norme ISO, ce problème est résolu en "regroupant" 3 échantillons si $L \leq 9$. Le nombre bits consommés est alors donné par

$$b = \frac{1}{3}\lceil \log_2 L^3 \rceil.$$

La quatrième colonne de la table 7.1 donne la résolution b (le nombre de bits équivalent par échantillon) en fonction de L.

La table 7.2 indique quels sont les quantificateurs permis dans chacune des sous-bandes. Ils sont indiqués par leur nombre de pas de quantification [27, page 52]. On remarque que les sous-bandes se décomposent en 5 groupes. Les deux premiers groupes, correspondant aux fréquences comprises entre 0 et 7.5 kHz, accepte 16 quantificateurs. Il faudra donc, en premier lieu, consacrer

[3] Il s'agit de quantificateurs de type "midtread" [2, page 117].

NoQ	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
SB0	0	3	7	15	31	63	127	255	511	1023	2047	4095	8191	16383	32767	65535
SB1	0	3	7	15	31	63	127	255	511	1023	2047	4095	8191	16383	32767	65535
SB2	0	3	7	15	31	63	127	255	511	1023	2047	4095	8191	16383	32767	65535
SB3	0	3	5	7	9	15	31	63	127	255	511	1023	2047	4095	8191	65535
SB4	0	3	5	7	9	15	31	63	127	255	511	1023	2047	4095	8191	65535
SB5	0	3	5	7	9	15	31	63	127	255	511	1023	2047	4095	8191	65535
SB6	0	3	5	7	9	15	31	63	127	255	511	1023	2047	4095	8191	65535
SB7	0	3	5	7	9	15	31	63	127	255	511	1023	2047	4095	8191	65535
SB8	0	3	5	7	9	15	31	63	127	255	511	1023	2047	4095	8191	65535
SB9	0	3	5	7	9	15	31	63	127	255	511	1023	2047	4095	8191	65535
SB10	0	3	5	7	9	15	31	63	127	255	511	1023	2047	4095	8191	65535
SB11	0	3	5	7	9	15	31	65535								
SB12	0	3	5	7	9	15	31	65535								
SB13	0	3	5	7	9	15	31	65535								
SB14	0	3	5	7	9	15	31	65535								
SB15	0	3	5	7	9	15	31	65535								
SB16	0	3	5	7	9	15	31	65535								
SB17	0	3	5	7	9	15	31	65535								
SB18	0	3	5	7	9	15	31	65535								
SB19	0	3	5	7	9	15	31	65535								
SB20	0	3	5	7	9	15	31	65535								
SB21	0	3	5	7	9	15	31	65535								
SB22	0	3	5	7	9	15	31	65535								
SB23	0	3	5	65535												
SB24	0	3	5	65535												
SB25	0	3	5	65535												
SB26	0	3	5	65535												

TAB. 7.2 – Nombre de pas de quantification possible par sous-bandes.

4 bits pour spécifier le numéro du quantificateur sélectionné. On notera ce numéro NoQ_k. Ces deux premiers groupes se distingue par une configuration différente des quantificateurs possibles. Le troisième groupe accepte 8 quantificateurs, le quatrième 4. Les sous-bandes 27 à 32 ne sont pas codées.

Pour reconstruire le signal, le récepteur a besoin de connaître non seulement les mots de code associés à chaque composante du vecteur $\underline{a}_k$ et aux facteurs d'échelle mais aussi l'allocation de bits réalisée à l'émetteur. Cette dernière information doit donc être transmise dans le flux binaire. Elle coûte $11 \times 4 + 12 \times 3 + 4 \times 2 = 88$ bits. En supposant par exemple que 20 sous-bandes soient transmises parmi les 27 possibles (les 5 dernières ne le sont jamais), l'information adjacente (allocation de bits + facteurs d'échelle) représente 88+120 bits. A 96 kbit/s, le nombre de bits restants pour coder les amplitudes des signaux de sous-bande normalisés est égal à $96 \times 384/44.1 - 88 - 120 = 628$ bits. A 64 kbit/s, il reste 350 bits. On voit donc que le nombre de bits réservé à la partie "noble" baisse beaucoup. On comprend alors pourquoi ce codeur fournit une bonne qualité tout en diminuant le débit jusqu'à ce que l'on atteigne un débit critique en dessous duquel la qualité s'effondre.

7.3 Codeur MPEG-2 AAC

Tous les détails techniques de ce codeur peuvent être trouvés dans [28]. Une description plus didactique est réalisée dans l'article [42]. On se contente ici d'en donner les principes.

On montre, figure 7.6, du signal de piano dans le domaine temporel (à gauche) et dans le domaine fréquentiel (à droite). Cette représentation spectrale est simplement obtenue en prenant les modules des composantes du vecteur $\underline{X}(m)$ et en les exprimant en dB. La transformée utilisée dans le codeur MPEG-2 AAC est la MDCT avec $N = 2048$ et $M = 1024$. Le tracé de la figure 7.7 représente toujours $|\underline{X}|$ mais en abscisse les fréquences sont ici exprimées en Bark et en ordonnée l'échelle est devenue linéaire. On a un effet de zoom dans les basses fréquences et les valeurs importantes sont fortement amplifiées.

On observera d'abord que la MDCT est une transformée à valeurs réelles mais pas nécessairement positives. Le codeur MPEG-2 AAC traite les signes à part[4]. La quantification du module des composantes du vecteur $\underline{X}(m)$ est réalisée de la façon suivante. Supposons que l'on connaisse

[4]En réalité, pas toujours ! On ne rentrera pas dans ce genre de détails ici. On ne parlera pas non plus de la façon dont sont codés les signes.

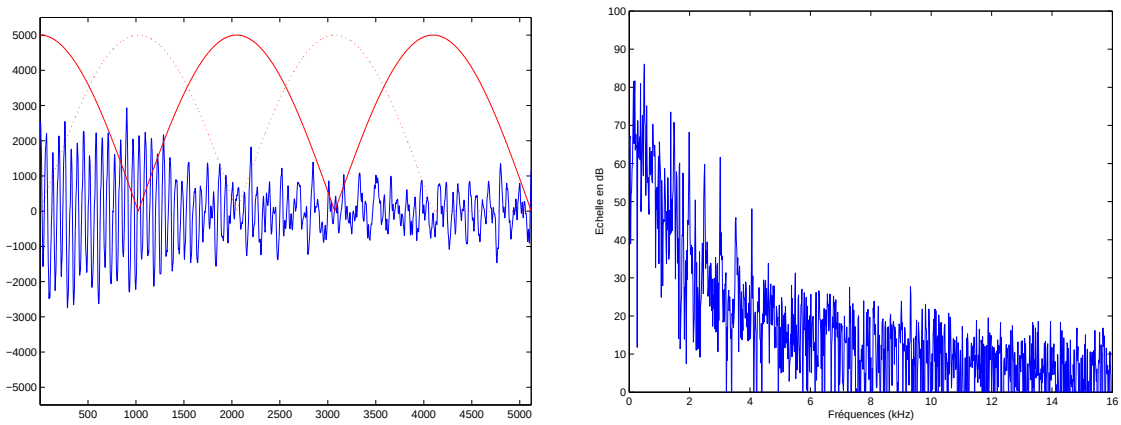

FIG. 7.6 – Signal de piano dans le domaine temporel (à gauche), dans le domaine fréquentiel (à droite)

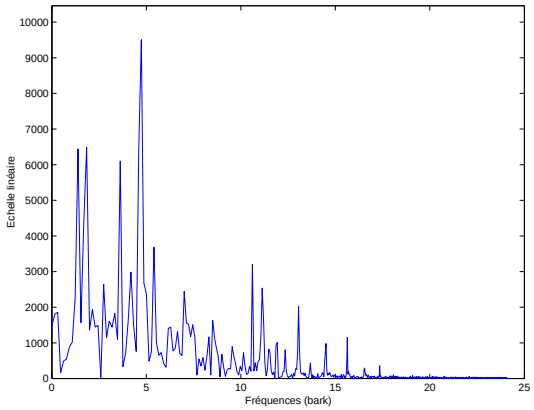

FIG. 7.7 – Modules des composantes du vecteur $\underline{X}(m)$ avec en abscisse les fréquences exprimées en Bark et en ordonnée une échelle linéaire.

M paramètres $\underline{g} = [g(0) \cdots g(M-1)]$ que l'on appellera par la suite "facteurs d'échelle". A l'émetteur, on calcule un vecteur d'entiers suivant la formule

$$\underline{i} = round([\frac{|X(0)|}{g(0)} \cdots \frac{|X(M-1)|}{g(M-1)}]).$$

Remarquons, qu'au récepteur, connaissant $\underline{i}$ et $\underline{g}$, on peut reconstruire

$$[|\hat{X}(0)| \cdots |\hat{X}(M-1)|] = [g(0) \times i(0) \cdots g(M-1) \times i(M-1)]$$

puis $[\hat{x}(0) \cdots \hat{x}(M-1)]$ en récupérant les informations de signe mais que l'on fait une erreur de reconstruction à cause de l'opération d'arrondi. Les tracés de la figure 7.8 montrent, pour un choix donné du vecteur des facteurs d'échelle $\underline{g}$ (courbe en marche d'escalier sur le tracé de gauche), le vecteur d'entiers $\underline{i}$ (à droite). On observera les valeurs en ordonnées. De valeurs réelles comprises entre 0 et 10000, on passe à des valeurs entières comprises entre 0 et 5.

Connaissant le vecteur $\underline{i} = [i(0) \cdots i(M-1)]$, on peut se poser un problème de codage et se demander quel est le coût en bits de cette information. Sur cet exemple, comme tous les entiers sont inférieurs à 8, 3 bits suffisent pour coder exactement chaque entier. Le nombre total de bits suffisant pour représenter exactement le vecteur $\underline{i}$ est donc égal à $M \times 3$. Ce nombre total

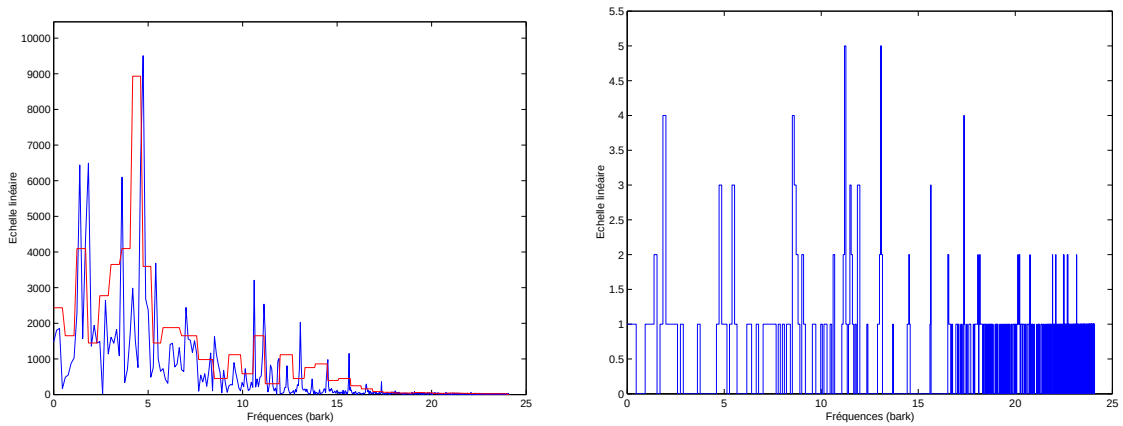

FIG. 7.8 – Les vecteurs $|\underline{X}|$ et $\underline{g}$ à gauche. Le vecteur $\underline{i}$ à droite.

est-il nécessaire ? Il existe certainement une solution plus économique en réalisant un codage de Huffman. En réalité, la solution adoptée dans ce codeur est de partitioner l'axe des fréquences en 51 "bandes" correspondant *grosso modo* à des demi-bandes critiques, de déterminer dans chaque bande la valeur maximale atteinte par $i(k)$ puis de réaliser un codage séparé dans chaque bande en utilisant des tables de Huffman indexées sur la valeur maximale et construite par apprentissage. La chaîne binaire devra donc comporter les mots de code des M entiers $i(k)$, les mots de code des 51 valeurs maximales et les mots de code des facteurs d'échelle $g(k)$.

Il ne reste plus qu'à déterminer le vecteur des facteurs d'échelle $\underline{g}$. *A priori* on peut voir ce problème comme un simple problème d'optimisation : il s'agit de déterminer $\underline{g}$ minimisant la puissance de l'erreur de reconstruction sous la contrainte que le nombre de bits nécessaire soit inférieur au nombre de bits disponibles. Cette solution n'est en fait pas réaliste car on peut constater qu'elle n'autoriserait que des taux de compression faibles (de l'ordre de 2) si on veut que le signal reconstruit soit transparent. Pour obtenir des taux de compression plus élevés (de 10 à 15), il faut exploiter les résultats de psychoacoustique et "mettre en forme spectralement" le bruit de reconstruction. Il s'agit, en réalité, d'un problème d'optimisation sous 2 contraintes : une contrainte de débit, le nombre de bits nécessaire doit être inférieur au nombre de bits disponible, et la contrainte "psychoacoustique" $S_Q(f) < \Phi(f)\ \forall f$. L'algorithme d'optimisation proposé dans le document normatif est un algorithme de type gradient basé sur la propriété suivante. Si on augmente un facteur d'échelle $g(k)$, on diminue l'entier $i(k)$. On diminue donc le débit mais on augmente la puissance du bruit de reconstruction dans la zone fréquentielle caractérisée par l'indice k. Rappelons que l'algorithme proposé est facultatif, il ne fait pas partie de la norme.

Comme les facteurs d'échelle doivent être transmis, ils consomment une partie du débit binaire disponible. Il n'est pas raisonnable de vouloir transmettre M facteurs d'échelle. La partition de l'axe des fréquences précédente en 51 bandes est aussi utilisées pour les facteurs d'échelle ce qui explique la courbe en marches d'escalier de la figure 7.8. Le codage du premier facteur d'échelle est réalisé sur 8 bits, le codage des 50 suivants est réalisé en codant $\Delta(k) = g(k) - g(k-1)$ par une table de Huffman.

Notons que le codeur MPEG-2 AAC est un codeur perceptuel "en boucle fermée". L'algorithme de détermination des facteurs d'échelle exploite la valeur réelle de l'erreur de reconstruction et non la valeur supposée après une opération d'arrondi.

Cette présentation ne donne que les grands principes. Une étude plus détaillée est nécessaire si on veut avoir une bonne connaissance de ce codeur. De nombreux points devraient être approfondis. Prenons juste un exemple et examinons le tracé de la figure 7.9. Le signal de musique visualisé est un signal de piano mettant en évidence la frappe d'une note. Il présente une mon-

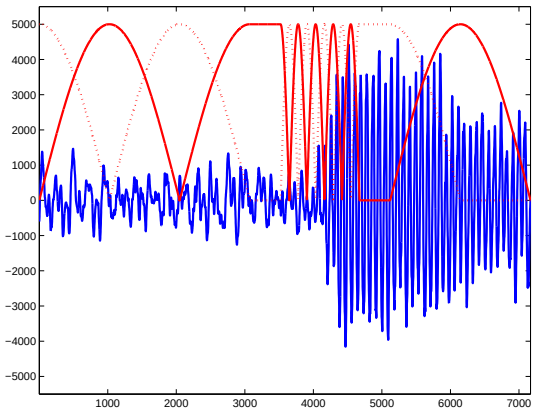

FIG. 7.9 – Commutation fenêtres longues/fenêtres courtes.

tée en puissance importante. Tant que le signal est relativement stationnaire dans une fenêtre d'analyse, on a sans doute intérêt à rechercher une bonne résolution fréquentielle ce qui nécessite une fenêtre d'analyse longue. Dans la ou les fenêtres où le signal présente une non stationnarité caractérisée, on a plutôt intérêt à rechercher une bonne résolution temporelle. On a alors besoin de fenêtres d'analyse plus courtes. Dans le codeur MPEG-2 AAC, une fenêtre de $N = 2048$ échantillons est divisée en 8 sous fenêtres de 256 échantillons. Pour respecter les conditions de reconstruction parfaite données par l'équation (3.2), il faut introduire des fenêtres de transition en amont et en aval. Bien entendu il faut définir un critère qui permette de décider une commutation fenêtres longues/fenêtres courtes. Le critère proposé[5] dans le document normatif est assez subtil. Il exploite le gain de prédiction : dans une montée en puissance le signal n'est pas très prédictible. Cette commutation fenêtres longues/fenêtres courtes donne de bons résultats mais à un effet très important sur le délai de reconstruction. En effet, on ne peut décider une commutation qu'après avoir observé complètement une montée en puissance et dans ce cas il faut introduire une fenêtre de transition en amont de la montée en puissance. On se rappelle que des délais de reconstruction longs sont incompatibles avec des communications bi-directionnelles ce qui explique l'existence d'une version du codeur AAC dite "low delay" dans MPEG-4, version dans laquelle la notion de fenêtres courtes a été supprimée ...

7.4 Codeur Dolby AC-3

Ce codeur [43] a une architecture très classique. La transformation temps-fréquences est réalisée par une MDCT avec des fenêtres recouvrantes de 512 échantillons. Les fenêtres deviennent deux fois plus petites si est observée une évolution significative de la puissance dans la deuxième moitié de la fenêtre. L'originalité de ce codeur est d'exploiter le format virgule flottante (mantisse + exposant) pour coder les coefficients de la transformée. Une opération d'arrondi est réalisée sur la mantisse avec une précision fonction du rapport signal à masque. L'exposant joue le rôle de facteur d'échelle. L'information spécifique de l'allocation de bits n'est pas transmise directement : elle est reconstruite au décodeur à partir de l'enveloppe spectrale qui est codée comme dans un codeur de parole.

[5]Bien entendu, il ne fait pas partie de la norme.

7.5 Modèle psychoacoustique : calcul du seuil de masquage

7.5.1 Introduction

On présente dans cette section, de façon la plus synthétique possible, quelques informations permettant de comprendre la signification du phénomène de masquage et comment est calculé un seuil de masquage. L'unique résultat de la psychoacoustique qui nous intéresse ici est de chercher à déduire des caractéristiques spectrales d'un signal de musique, la densité spectrale d'un bruit susceptible d'être rajouté au signal original sans que ce bruit soit perçu. Pour plus d'informations concernant le domaine de la perception des sons, on consultera par exemple [44, 45, 46].

7.5.2 L'oreille

On sait que l'oreille se décompose en trois parties : l'oreille externe qui est formée du pavillon et du conduit auditif, l'oreille moyenne comprenant la chaîne ossiculaire et le tympan et l'oreille interne comprenant la cochlée. L'oreille externe et l'oreille moyenne réalisent une fonction de filtrage de type passe-bande entre 20 Hz et 20 kHz dont le module de la réponse en fréquence est directement relié au seuil d'audition absolu. Elles réalisent aussi une fonction d'adaptation d'impédance entre le milieu aérien de propagation des ondes et le milieu liquide de l'oreille interne. L'oreille interne est composée de la cochlée qui contient un liquide dans lequel baigne la membrane basilaire. Celle-ci est stimulée par le mouvement de l'étrier. Cette membrane peut être vue, une fois déroulée, comme un segment de droite de longueur donnée (3.5 cm) d'où part un grand nombre de fibres nerveuses. Toute excitation sonore met en vibration cette membrane. Tout se passe comme si un son pur (une sinusoïde) à une fréquence donnée f_1 et de puissance σ_X^2 voyait cette puissance "étalée" sur une portion de la membrane basilaire centrée autour d'une position particulière fonction de f_1 et comme si chaque fibre mesurait cette puissance étalée en délivrant en conséquence des "décharges de potentiel d'action" dans les neurones du nerf auditif. Si on appelle k la position d'une fibre particulière le long de la membrane basilaire et $S_E(k)$ la quantité de puissance mesurée par cette fibre, Zwicker [44] déclare légitime d'interpréter k comme une fréquence reliée à la fréquence f_1 par une relation non-linéaire. Celle-ci devient linéaire si la fréquence est exprimée suivant une nouvelle échelle appelée *échelle des Bark*. Cette nouvelle échelle des fréquences est directement reliée à la notion de bandes critiques présentée à la sous-section 7.5.3. Les "fonctions d'étalement" $S_E(k)$ qui seront à l'origine du calcul du seuil de masquage comme on le verra à la sous-section 7.5.4, reflètent l'étendue et l'importance de l'excitation créée dans le système auditif par le son excitant.

Zwicker a montré que l'hypothèse qu'il avait faite conduisait à prendre un banc de filtres comme représentation du système auditif (en oubliant le filtrage passe-bande initial). En effet, imaginons une sinusoïde $x(n) = a\ sin(2\pi f_1 n)$ entrant dans M filtres de réponse en fréquence $H_k(f)$. Les M signaux de sous-bandes ont pour expression

$$y_k(n) = a|H_k(f_1)|sin(2\pi f_1 n + \phi_k(f_1)).$$

Ils ont comme puissance $\sigma_{Y_k}^2 = \sigma_X^2|H_k(f_1)|^2$. Le terme $\sigma_{Y_k}^2$ représente la puissance transmise par le son pur excitant de fréquence f_1 à la $k$ème position de la membrane basilaire. Si on trace $|H_k(f_1)|^2$ en fonction de k, on obtient la fonction d'étalement $S_E(k)$ fonction de f_1.

7.5.3 Bandes critiques

Différentes expériences conduisent à la même notion de bandes critiques et fournissent approximativement les mêmes largeurs de bandes. On ne décrit ici qu'une seule, celle qui est directement reliée au seuil d'audition absolu. Deux sons purs simultanés de fréquences différentes sont détectés à une puissance plus faible que ne l'est un seul d'entre eux à la condition que leur écart

de fréquence ne dépasse pas une certaine valeur. Plus précisément, considérons une sinusoïde à la fréquence f_1. Si la puissance σ_X^2 de cette sinusoïde vérifie $\sigma_X^2 \geq S_a(f_1)$ où $S_a(f)$ est le seuil d'audition absolu, cette sinusoïde sera audible. Imaginons une deuxième sinusoïde de fréquence f_2 voisine de f_1 et de puissance identique à la première. Des mesures expérimentales montrent que l'ensemble des deux sinusoïdes sera audible pourvu que $2\sigma_X^2 \geq S_a(f_1)$. Considérons N sinusoïdes à des fréquences $f_1 < f_2 < \cdots < f_N$ régulièrement réparties et de même puissance σ_X^2. La propriété précédente, à savoir l'ensemble des N sinusoïdes est audible pourvu que $N\sigma_X^2 \geq S_a(f_1)$ est vérifiée à la condition que la largeur de bande $\Delta f = f_N - f_1$ soit inférieure à un seuil appelé largeur de la bande critique dans le voisinage de la fréquence f_1. Cette expérience montre que, finalement, on peut admettre que la puissance perçue par l'oreille dans une bande critique est égale à la somme de toutes les puissances des composantes dans cette bande de fréquence : l'oreille réalise une intégration des puissances dans une bande critique. On peut également dire que l'oreille se comporte comme un banc de filtres réalisant une partition non-régulière de l'axe des fréquences puis somme la puissance dans chaque sous-bande.

La bande audible est divisée en 24 bandes critiques dont la répartition fréquentielle est donnée table 7.3 [44]. On définit une nouvelle échelle fréquentielle, l'échelle des Barks[6], correspondant simplement au numéro de la bande critique.

Numéro de la bande	Fréquence inférieure	Fréquence supérieure	Largeur de la bande critique
1	20	100	80
2	100	200	100
3	200	300	100
4	300	400	100
5	400	510	110
6	510	630	120
7	630	770	140
8	770	920	150
9	920	1080	160
10	1080	1270	190
11	1270	1480	210
12	1480	1720	240
13	1720	2000	280
14	2000	2320	320
15	2320	2700	380
16	2700	3150	450
17	3150	3700	550
18	3700	4400	700
19	4400	5300	900
20	5300	6400	1100
21	6400	7700	1300
22	7700	9500	1800
23	9500	12000	2500
24	12000	15500	3500
25	15500		

TAB. 7.3 – Bandes critiques. Les fréquences sont exprimées en Hz.

Dans cette table, les 24 bandes critiques sont artificiellement juxtaposées. En fait, l'oreille peut former une bande critique autour de n'importe quelle fréquence. La relation entre une fréquence exprimée en Hertz dans l'intervalle [20 Hz - 20 kHz] et une fréquence exprimée en Bark

[6]Il existe une autre échelle, l'échelle MEL, qui est approximativement 100 fois plus fine.

dans l'intervalle [1 - 24] s'approxime par l'expression suivante

$$f_B = 13 \times arctg[0.76\frac{f_H}{1000}] + 3.5 \times arctg[(\frac{f_H}{7500})^2].$$

7.5.4 Courbes de masquage

L'oreille a la propriété de ne pas percevoir un son en présence d'un autre pourvu que les caractéristiques de ces deux sons aient des relations particulières. C'est le *phénomène de masquage*. Il existe deux types de masquage d'un son par un autre. On parle de masquage temporel si deux sons apparaissent de façon successive et de masquage fréquentiel si deux sons apparaissent simultanément. Le phénomène de masquage temporel peut être intéressant en codage mais il ne sera pas présenté ici. Introduisons le masquage fréquentiel.

On rappelle qu'une sinusoïde de fréquence f_1 n'est audible, dans une ambiance parfaitement silencieuse, qu'à la condition que sa puissance soit supérieure au seuil d'audition absolu à cette fréquence. Considérons maintenant le cas de deux sinusoïdes. La première sinusoïde, la sinusoïde masquante, est à une fréquence particulière f_1 et a une puissance $\sigma^2_{X_1}$ donnée, largement supérieure à $S_a(f_1)$. On mesure, pour toutes les fréquences f_2 dans la bande audible, la puissance $\sigma^2_{X_2}$ de la deuxième sinusoïde, la sinusoïde masquée, à la limite d'audition en présence de la première sinusoïde. La fonction $\sigma^2_{X_2}(f_2)$ s'appelle la courbe de masquage ou la courbe d'effet de masque. L'allure de cette courbe de masquage est donnée figure 7.10 (à gauche) lorsque le son masquant est une sinusoïde.

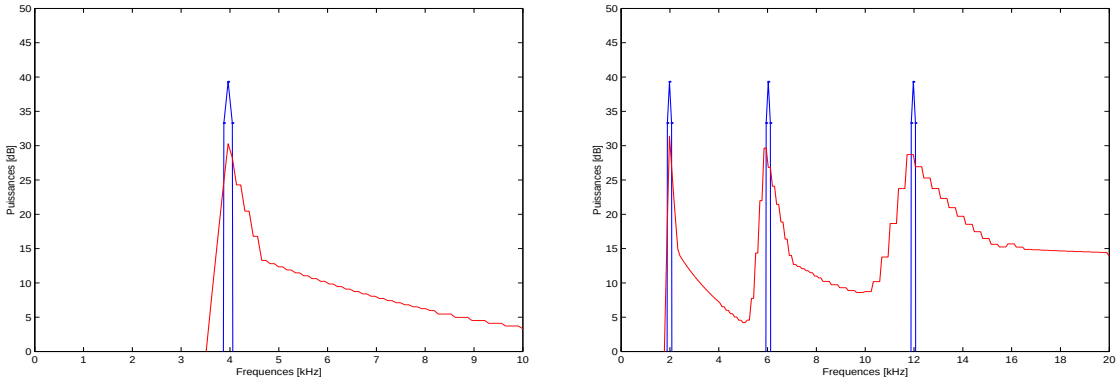

FIG. 7.10 – A gauche : courbe de masquage lorsque le son masquant est une sinusoïde. A droite : influence de la fréquence sur la forme de la courbe de masquage.

Les courbes de masquage d'une sinusoïde par une autre sinusoïde[7] ne sont pas les seules courbes intéressantes. La psychoacoustique nous donne aussi des résultats précis dans le cas où le signal masqué est un bruit à bande étroite. De plus, un signal musical pouvant être considéré comme composé d'un certain nombre de sons purs, modélisables par des sinusoïdes, et de sons qui ne le sont pas, modélisables par des bruits à bande étroite, il est nécessaire d'examiner les quatre cas suivants : le masquage d'une sinusoïde par une sinusoïde, le masquage d'une sinusoïde par un bruit à bande étroite et plus particulièrement le masquage d'un bruit à bande étroite par une sinusoïde et le masquage d'un bruit à bande étroite par un bruit à bande étroite. Les courbes représentées figure 7.10 sont, en réalité, les courbes de masquage d'une sinusoïde par un bruit à bande étroite et ont été obtenues par simulation en appliquant le modèle psychoacoustique no 1 de MPEG.

[7]Dans la littérature, on appelle courbe d'effet de masque d'un son X le seuil d'audition relevé en présence du son X masquant. Pour éviter toute ambiguïté, on préfère parler ici de courbe de masquage d'un son Y par un son X.

Toutes ces courbes, quel que soit le cas envisagé[8] ont la même allure, une forme en triangle. Examinons ces courbes de façon plus précise et étudions l'influence des paramètres f_1 et $\sigma^2_{X_1}$ sur $\sigma^2_{X_2}(f_2)$. La première propriété de ces courbes est de présenter leur maximum à la fréquence f_1. On remarque sur la figure 7.10 que la puissance $\sigma^2_{X_2}$ à la fréquence f_1 est inférieure à la puissance $\sigma^2_{X_1}$. La différence $\sigma^2_{X_2}(f_2 = f_1) - \sigma^2_{X_1}$ s'appelle *l'indice de masquage*. La deuxième propriété est de présenter une dissymétrie importante. La pente des courbes est plus accentuée vers les fréquences inférieures à la fréquence du son masquant que vers les fréquences supérieures. La pente des courbes dépend de la fréquence f_1 du son masquant. Elle devient moins accentuée à mesure que la fréquence f_1 croît. Si l'on exprime la fréquence en Bark et les puissances en dB, on montre que les courbes de masquage peuvent être modélisées par des segments de droite dont la pente devient indépendante de la fréquence f_1. Hélas, la pente vers les fréquences supérieures reste dépendante de $\sigma^2_{X_1}$. Elle est d'autant moins accentuée que $\sigma^2_{X_1}$ est important.

7.5.5 Seuil de masquage

Le développement précédent est encore insuffisant pour nous car il ne traite que le cas où le son masquant et le son masqué sont des sons purs ou des bruits à bande étroite de largeur de bande inférieure ou égale à celle d'une bande critique. D'une part, un signal audio est composée d'une infinité de contributions : la densité spectrale de puissance $S_X(f)$ caractérise justement l'importance relative de chaque contribution. D'autre part, on cherche à déterminer un signal à la limite d'inaudibilité, le bruit de quantification, et ce signal possède lui aussi une infinité de contributions.

Deux types de problèmes se posent. On est en présence de nombreux sons masquants : comment faut-il les "additionner" pour déterminer la puissance du son masqué juste à la limite d'audition ? On est en présence de nombreux sons masqués : comment peut-on généraliser ? Ces problèmes sont difficiles ; ils sont peu traités dans la littérature écrite par les acousticiens parce qu'ils sont spécifiques de l'application codage[9]. Les psychoacousticiens ne savent pas décrire avec précision le phénomène de masquage lorsque le nombre de composantes masquantes est supérieur à 2 ou 3 alors que l'on a besoin de pouvoir traiter le cas de 512 ou même 1024 composantes masquantes ! C'est tout le problème des "lois d'addition" en psychoacoustique comme on vient de le mentionner. Les modèles disponibles en codage ont été déterminés par des ingénieurs qui ont défini la valeur numérique de quelques paramètres après un très grand nombre d'écoutes mettant en œuvre le codeur complet.

Un exemple de seuil de masquage dans la bande $[0 - 10]$ kHz calculé à partir d'un signal de violon est donné à la figure 7.3. Il s'agit du modèle psychoacoustique no 1 du standard MPEG-1 fourni de façon facultative dans le document normatif.

[8] excepté le masquage d'une sinusoïde par une sinusoïde, car il introduit des phénomènes de battement lorsque deux fréquences de sons purs sont proches, mais il est inutile en codage

[9] et depuis peu de temps de l'application tatouage

Chapitre 8

Codage audio : compléments

Le développement de codeurs perceptuels de haute qualité à débit réduit a été l'objet d'importants efforts de recherche ces vingt dernières années. Le codeur actuellement le plus abouti est le codeur AAC présent dans MPEG-2 et MPEG-4. Il fait toujours l'objet d'améliorations dues essentiellement à l'introduction de nouveaux outils. Dans ce chapitre on présente deux évolutions récentes caractéristiques : la recherche de codeurs de bonne qualité à faible débit mais aussi la recherche de codeurs de très bonne qualité voir du sans perte. Notons toutefois que ces deux évolutions ne présentent pas du tout les mêmes taux de compression et par conséquent le même intérêt. Disposer d'un codeur à bas débit tout en maintenant une qualité acceptable intéresse considérablement plus les opérateurs que de disposer d'un codeur sans perte présentant un faible taux de compression. Un développement sera quand même consacré à ce codeur dans ce livre car il possède une propriété intéressante sur le plan pédagogique : les algorithmes utilisés dans ce codeur sont des adaptations directes d'algorithmes développés en codage d'image. Cette convergence codage audio/codage d'image est suffisamment rare pour être notée.

Dans ce chapitre, on présentera relativement sommairement ces évolutions récentes du codeur AAC. Dans le chapitre suivant, on réalisera une étude beaucoup plus approfondie de l'outil consacré au codage stéréo.

8.1 Codeurs bas-débits/qualité acceptable

Le codeur audio HE-AACv2 développé dans le cadre de MPEG-4 et de l'ETSI 3rd Generation Partnership project (3GPP) est construit autour du codeur MPEG-4 AAC et exploite les deux outils intitulés "Spectral Band Replication" (SBR) et "Parametric stereo" (PS). Il a pour objectif la gamme de débits 8 kbit/s mono à 48 kbit/s en stéréo (ou au format 5.1 ou "surround" à 64 kbit/s).

8.1.1 Premier outil : "Spectral Band Replication" (SBR)

Cette technique d'extension de bande a été proposée en 1999 dans le cadre de la normalisation de la radio numérique en bande AM (consortium "Digital Radio Mondiale" (DRM)). Elle a été normalisée en 2003. Elle consiste à chercher à reconstruire à chaque instant le contenu hautes fréquences du signal à partir de son contenu basses fréquences. Par exemple, pour un son périodique entretenu, le signal est composé d'une fréquence fondamentale et d'une série d'harmoniques dont les fréquences sont multiples de la fréquence fondamentale comme le suggère le tracé de la figure 8.1. La position des harmoniques suivant l'axe des fréquences n'a donc pas besoin d'être transmise. Il suffit de coder l'amplitude de ces harmoniques. Cela est réalisé en général en codant l'enveloppe spectrale calculée, par exemple, par une "analyse LPC". On a vu que le codage de l'enveloppe spectrale, pour un codeur de parole, nécessite de l'ordre d'un kbit/s. La

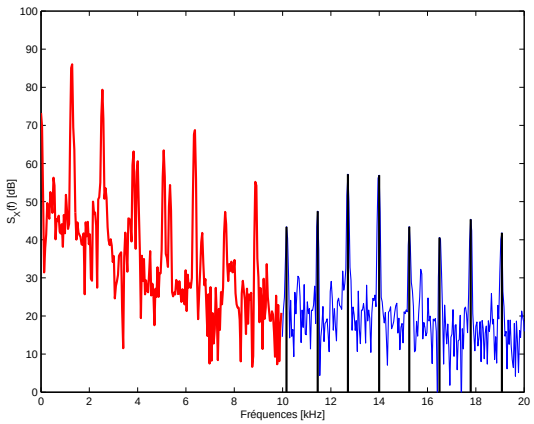

FIG. 8.1 – Possibilité de reconstruire le spectre hautes fréquences à partir des basses fréquences.

majeure partie du débit disponible est donc réservée au codage du contenu basses fréquences, la partie hautes fréquences ne réclamant globalement que quelques kbit/s.

Le principe de cette méthode est le suivant.

- Le signal d'entrée $x(n)$ est sous-échantillonné, par exemple par un facteur 2, puis codé par un codeur standard fournissant la 1ère partie de la chaîne binaire. Si ce codeur donne une qualité raisonnable à un débit de D bit/s lorsqu'il est appliqué à des signaux pleine bande, on peut raisonnablement penser qu'il nécessitera $D/2$ bit/s lorsqu'il exploitera des signaux demi-bande. Dans le cadre du codeur audio HE-AACv2, il s'agit bien entendu du codeur AAC. Le codeur AAC mis en œuvre est pratiquement le codeur AAC habituel excepté peut être le fait que le modèle psychoacoustique a été modifié étant données les exigences réduites de ce codeur.
- Le signal d'entrée est aussi traduit dans le domaine fréquentiel et analysé. Dans le cadre du codeur audio HE-AACv2, le signal d'entrée est traité par un banc de 64 filtres QMF fournissant des signaux de sous-bande. La 2ème partie de la chaîne binaire code des informations caractéristiques de l'enveloppe spectrale, de la présence ou l'absence d'harmoniques et de la relation entre puissance des composantes tonales et non tonales.

8.1.2 Deuxième outil : "Parametric stereo" (PS)

Un bref historique

Dès que le problème de la compression audio s'est posé, c'est à dire au milieu des années 80, plusieurs techniques ont été développées pour coder des signaux en stéréo. La démarche initiale a consisté à réaliser un codage perceptuel sur chacun des deux canaux mais ce codage séparé des deux canaux entraînant des débits prohibitifs, rapidement on a vu deux techniques s'imposer.

- Le "M/S stereo coding" d'abord. Au lieu de transmettre la voie gauche et la voie droite on transmet la somme normalisée (M : middle channel) et la différence (S : side channel). Ce matriçage est réalisé dans le domaine fréquentiel de façon sélective. Ce procédé est inversible (c'est un avantage) mais le taux de compression est fonction du signal (c'est un inconvénient). Le taux de compression est au maximum égal à 2 quand les deux canaux gauche et droite sont pratiquement égaux mais ce n'est pas le cas le plus fréquent. Pour des signaux audio multicanaux, ce procédé peut être utilisé par paire d'enceintes.
- L'"Intensity stereo coding" ensuite. Au dessus de 2 kHz, le système auditif humain n'est pas sensible aux différences de phase entre les signaux gauche et droite d'un enregistrement stéréo. La technique "intensity stereo" exploite cette propriété en ne transmettant dans

les hautes fréquences qu'un seul signal plus des facteurs d'échelle fonctions du temps et des fréquences pour encoder des différences d'intensité entre les deux voies stéréo. On exploite le fait que la perception des composantes hautes fréquences est directement reliée à l'enveloppe énergétique temporelle. En considérant que ce codage est effectué dans la demi-bande hautes fréquences, le gain en débit est de l'ordre de 25 % pour des signaux stéréo. Le gain peut être supérieur pour des signaux multi-canaux car il n'y a toujours transfert que d'un seul canal (dans la demi-bande hautes fréquences).

Un regain d'intérêt s'est exprimé ces dernières années pour la compression audio multicanaux (stéréo mais aussi pour le format 5.1) sous la pression du marché du DVD ou des opérateurs téléphoniques qui veulent, par exemple, des systèmes de reproduction du son de bonne qualité sur un téléphone mobile ...

Principe du "codage audio paramétrique stéréo"

Le schéma de principe est donné à la figure 8.2. Lorsque le débit visé est faible (quelques

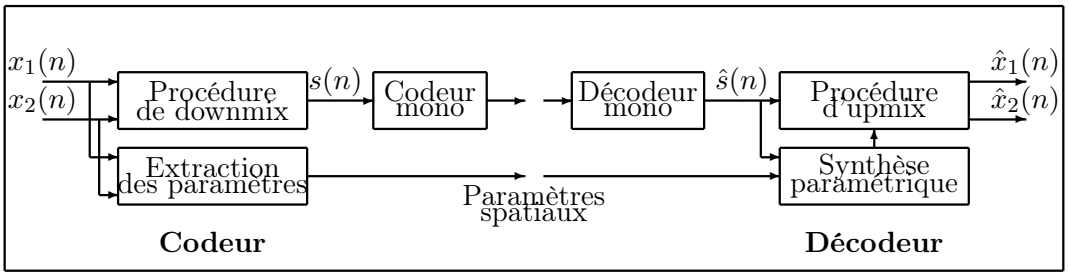

FIG. 8.2 – Schéma de principe d'un codeur audio paramétrique

dizaines de kbit/s), il n'existe plus guère qu'une seule possibilité : construire d'abord à partir de tous les canaux un seul signal (procédure de "downmix"), coder ce signal avec un codeur standard, par exemple le codeur MPEG-4 AAC, et transmettre cette information avec des informations adjacentes. Le développement de la sous section suivante consacrée à la perception de l'espace sonore montrera qu'en réalité il existe trois informations adjacentes significatives : la différence intercanale de temps (ICTD), la différence intercanale d'intensité (ICLD) et la cohérence intercanale (ICC). On peut interpréter cette démarche comme une généralisation de la technique "Intensity stereo". Au récepteur, on reconstruit une image spatiale en fabriquant un signal stéréo ou multicanaux (procédure d'"upmix") à partir du signal mono codé et des informations adjacentes. Ces informations adjacentes étant indépendantes du codeur, cette technique est assez générale. Elle présente aussi le grand intérêt d'assurer une compatibilité arrière complète avec une transmission mono. On peut remarquer aussi qu'il n'est pas nécessaire de s'intéresser aux conditions de masquage dans un contexte binaural.

Deux équipes distinctes ont particulièrement travaillé dans ce domaine en proposant deux techniques assez similaires. Faller *et al.* dénomment leur proposition "codage par indices binauraux" (BCC pour Binaural Cue Coding) [47] [48] [49]. Breebart *et al.* parlent de "codage stéréo paramétrique"' (PS pour Parametric Stereo) [50]. D'autres équipes proposent des contributions, par exemple [51] [52]. On pourra trouver des références aux travaux MPEG dans [53] [54].

Résultats

La description paramétrique du champ sonore spatial qui est réalisée permet d'obtenir une représentation extrêmement compacte (quelques kbit/s). Cette technique a été développée dans le cadre de MPEG-4, spécialement pour le codeur HE-AACv2 pour obtenir un débit total de 24 kbit/s [54]. Ce procédé permet donc un taux de compression extrêmement élevé. Des tests

d'écoutes [54] montre que ce codeur à 24 kbit/s est approximativement équivalent au codeur HE-AAC stéréo à 32 kbit/s qui est lui même approximativement équivalent à l'AAC stéréo à 64 kbit/s. La qualité n'est pas parfaite mais acceptable. On parle de qualité "intermédiaire".

Récemment cette technique a été généralisée au format 5.1. Il s'agit du codeur MPEG Surround. On peut dire que le débit nécessaire pour obtenir une "bonne qualité" pour le format 5 canaux était de l'ordre de 300 kbit/s en 1994 avec le codeur MPEG-2 layer 3, de 200 kbit/s en 2000 avec le codeur MPEG-4 AAC, de 128 kbit/s en 2004 avec le codeur HE-AAC et aujourd'hui de 64 kbit/s avec cette nouvelle technologie de codage audio paramétrique.

8.1.3 Perception de l'espace sonore

On donne juste quelques résultats de psychoacoustique relatifs à l'écoute binaurale dans le cas le plus simple, une source sonore en champ libre [46, 55].

La direction d'incidence du son est caractérisée par son azimut (angle que fait la direction incidente avec le demi-plan vertical situé dans l'axe du visage) et l'élévation (angle que fait cette direction incidente avec un plan horizontal). La localisation en élévation qui donne des résultats très différents suivant la nature du signal et la localisation en distance qui est liée à l'intensité perçue (pour un son pur la distance subjective est d'autant plus importante que son intensité est élevée) n'interviendront pas par la suite.

Deux indices binauraux sont impliqués dans la localisation en azimut.

- La différence de temps d'arrivée d'un son à l'oreille gauche et à l'oreille droite : la différence interaurale de temps (ITD : Interaural Time Difference). Elle est significative surtout en basses fréquences, fréquences inférieures à 1500 Hz (longueur d'onde égale à 23 cm correspondant à la distance interaurale).
- L'atténuation due à la tête entraînant une différence d'intensité : la différence interaurale d'intensité (ou de niveau ILD : Interaural Level Difference). Une source placée à la gauche d'un auditeur engendre une plus grande intensité du signal à l'oreille gauche qu'à l'oreille droite. Elle est significative surtout en hautes fréquences.

Le troisième indice binaural exploité en codage est celui qui est relié à l'étendue subjective d'une source : un événement auditif peut être perçu de façon très ponctuelle ou au contraire être très peu localisé comme par exemple lors d'un concert dans une église à cause des réflexions multiples. On parle alors de cohérence interaurale (IC : Interaural coherence).

Il existe d'autres éléments de psychoacoustique relatifs à la perception de l'espace sonore mais ils ne sont guère exploités en codage audio. On en donne quelques exemples :

- Effet de précédence ou la loi du premier front d'onde. Deux sons qui arrivent aux oreilles dans un temps assez bref sont entendus comme un seul et c'est alors le premier qui détermine la perception spatiale.
- Si le délai est supérieur à quelques ms, les réflexions d'une source sonore sur les obstacles environnants (lorsqu'elles sont perçues distinctement) sont appelés des échos tandis que les réflexions suivantes se fusionnent habituellement pour former la réverbération.
- Le signal audio entrant dans le système auditif peut être vu comme une version filtrée du signal issu de la source. Ce filtrage est caractérisé par une fonction de transfert (HRTF : Head-Related Transfert Functions) qui dépend de chaque source. L'ensemble de la tête et du pavillon réalise un filtrage complexe dont l'effet est notable entre 500 et 16000 Hz.

Les résultats de psychoacoustique fournissent donc des indices relatifs aux signaux en entrée du système auditif. Dans une écoute au casque, les indices mesurés directement sur les signaux audio sont pratiquement identiques aux indices binauraux. Il n'en n'est plus de même *a priori* pour une retransmission par hauts-parleurs car il faudrait tenir compte des fonctions de transfert acoustiques du couple transducteurs/oreilles (HRTF). En réalité, dans une application de compression, le but n'est pas de chercher à interpréter finement le rôle de ces indices dans la

construction de l'effet spatial mais de faire en sorte que les indices relatifs aux signaux reconstruits au récepteur soient approximativement les mêmes que ceux des signaux originaux. Comme on évalue des indices similaires à l'ITD, ILD et IC directement sur les signaux source, il est préférable de définir de nouvelles appellations. On les appelle la différence intercanale de temps (ICTD), la différence intercanale d'intensité (ICLD) et la cohérence intercanale (ICC). Pour une application en compression, ce qui compte c'est que les trois paramètres intercanaux soient encodés et reconstruits correctement ce qui entraînera la similarité des paramètres binauraux des signaux originaux et reconstruits.

8.2 Codeurs haut-débits/sans perte ou presque sans perte

8.2.1 Introduction

Un codage sans perte n'entraîne jamais des taux de compression supérieurs à 2 ou 3. Des études montrent qu'un taux de compression moyen sur un corpus important est de l'ordre de 2 pour des signaux échantillonnés à 48 kHz et quantifiés sur 16 bits, légèrement supérieur pour des signaux échantillonnés à 96 kHz, légèrement inférieurs pour des signaux quantifiés sur 24 bits. Ce type de codage est essentiellement nécessaire pour des applications professionnelles (post production) où on ne peut pas se permettre de décoder un fichier de musique au format AAC, le traiter puis le sauvegarder au format AAC et de recommencer. Plusieurs encodage/décodage successifs (tandem coding) dégradent assez rapidement (au delà de 2 ou 3 tandem) le signal. Si on rajoute à la couche AAC (à 128 kbit/s en stéréo) une couche d'amélioration de 384 kbit/s, on n'observe plus de dégradation sensible avant un nombre important d'étapes d'encodage/décodage. Les principales applications sont donc des opérations en studio et pour de l'archivage.

Jusqu'à une date assez récente [56], les systèmes de compression sans perte disponibles (AudioPak, LTAC, Musiccompress ...) utilisaient le plus souvent un modèle prédictif de la forme

$$y(n) = x(n) - Q[\sum_{k=1}^{P} \hat{b}_k x(n-k) - \sum_{k=1}^{Q} \hat{a}_k y(n-k)]$$

où l'opération indispensable de quantification $Q(.)$ permettait de ramener $y(n)$ au même format que $x(n)$ et où le calcul des coefficients a_k et b_k était réalisé par des algorithmes classiques, par exemple l'algorithme de Levinson en virgule fixe. Ces systèmes de compression réalisaient ensuite un codage entropique de l'erreur de prédiction par codage de Huffman mais le plus souvent par codage de Rice qui est adapté à une source laplacienne c'est à dire une source où les faibles valeurs sont les plus fréquentes.

Vu l'importance de ce type de codage dans ces codeurs, on donne le principe du code de Rice à partir d'un exemple. Prenons l'exemple du codage de la valeur +26 qui a pour représentation binaire $0\cdots011010$, les points de suspension traduisant le fait qu'il faudrait une information supplémentaire pour être précis à savoir la valeur maximale permise. On choisit un entier m par exemple 3. Le mot de code est alors construit de la façon suivante : on code d'abord son signe, ici avec le bit 0, puis on utilise directement les 3 bits les moins signifiants i.e. 010. Il reste à représenter les bits les plus signifiants 11. Traduit de binaire en décimale cela donne 3, on rajoute donc 3 zéros suivi d'un 1 final ce qui donne finalement le mot de code 00100001. La connaissance de la valeur maximale possible de la source n'est pas nécessaire ici. Il faut évidemment que les valeurs faibles aient des probabilités d'apparition fortes pour que cela soit intéressant. Le paramètre m est constant dans une fenêtre d'analyse et peut être pris égal à $m = \log_2(\log_e(2)\mathbb{E}\{Y(n)\})$. Le code de Rice est un cas particulier du code de Golomb lorsque le paramètre m est une puissance de 2, ce qui facilite beaucoup son implantation.

8.2.2 Normalisation ISO/IEC MPEG-4

Principe

Un codeur sans perte ou scalable avec une granularité assez fine entre le sans perte et le codage AAC à un débit de l'ordre de 64 kbit/s par voie a été normalisé récemment [57], [58]. Le principe du traitement est le suivant. Le signal d'entrée est d'abord codé AAC (core layer), la différence entre le signal d'entrée et le signal reconstruit est ensuite codée de façon hiérarchique jusqu'au codage sans perte et fournit la couche d'amélioration (enhancement layer). Les deux couches sont ensuite multiplexées.

Concernant la couche d'amélioration, une transformation temps/fréquence sur le signal d'entrée est réalisée par une IntMDCT consistant en une approximation entière inversible de la MDCT. La soustraction avec ce qui est produit par le codeur AAC est réalisée dans le domaine fréquentiel. Un encodage de l'erreur par deux codages par plan de bits a lieu ensuite : un codage de Golomb et un codage arithmétique "codé context". Il y a pseudo-exploitation d'un modèle psychoacoustique en partant des MSB vers les LSB : l'erreur de reconstruction doit garder une forme spectrale approximativement équivalente au seuil de masquage lorsque l'on augmente le débit et que l'on se rapproche du codage sans perte.

Quelques détails

Remarquons d'abord que la transformée MDCT est décomposable en un ensemble de rotations élémentaires, chaque rotation étant caractérisée par une matrice de rotation, elle-même décomposable de la façon suivante

$$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} = \begin{bmatrix} 1 & \alpha \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ \beta & 1 \end{bmatrix} \begin{bmatrix} 1 & \alpha \\ 0 & 1 \end{bmatrix}$$

avec $\alpha = (\cos\theta - 1)/\sin\theta$ et $\beta = \sin\theta$. La transformation

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

se calcule alors en 3 étapes successives

$$z = x + [\alpha y], \quad Y = y + [\beta z], \quad X = z + [\alpha Y]$$

qui s'inversent directement

$$z = X - [\alpha Y], \quad y = Y - [\beta z], \quad x = z - [\alpha y].$$

Si à chaque étape on rajoute une opération d'arrondi, on obtient une transformation qui, à partir de valeurs entières, donne des valeurs entières et qui est inversible

$$\hat{z} = x + Q[\alpha y], \quad \hat{Y} = y + Q[\beta \hat{z}], \quad \hat{X} = \hat{z} + Q[\alpha \hat{Y}].$$

En exploitant cette propriété basée sur le schéma lifting proposé par Daubechies et Sweldens en 1996 ou sur les réseaux en échelle (Bruekers, Enden 1992), on obtient la IntMDCT. La différence entre les spectres de puissance de signaux de musique calculés à partir de la MDCT et de la IntMDCT montrent que la IntMDCT est une bonne approximation : la différence est blanche, une soixantaine de dB moins puissante que le signal original. Des résultats récents montrent que l'on peut rendre cette erreur plus puissante dans les basses fréquences que dans les hautes fréquences.

Concernant la soustraction dans le domaine fréquentiel, le calcul de l'erreur résiduelle entre les valeurs $c(k)$ du spectre calculé à partir de l'IntMDCT et leurs contreparties $\hat{c}(k)$ issues du

codeur AAC nécessite de prendre des précautions car il faut assurer l'"inversibilité" du codeur AAC. Les valeurs prises par $\hat{c}(k)$ doivent être entières et si on appelle $i(k)$ l'entier associé au mot de code spécifié par le codeur AAC à l'indice k, la relation entre $i(k)$ et $\hat{c}(k)$ doit être déterministe (utilisation de tables et interpolation).

Le codage entropique exploite le code de Golomb (adapté à des sources laplaciennes) par plan de bits (du MSB vers LSB pour créer une chaîne binaire imbriquée). L'utilisation d'un codage arithmétique est basé sur le contexte (décomposition des données spectrales en 3 bandes (<4 kHz <11 kHz).

Chapitre 9

Codage stéréo : une présentation synthétique

Au chapitre précédent on a juste donné le principe du "codage audio paramétrique stéréo". Dans ce chapitre, on se propose de réaliser une étude beaucoup plus précise de ce type de codage. L'article de Breebart [50] est une excellente introduction mais la lecture de cet article n'est pas toujours très facile parce que certains résultats importants sont souvent présentés sans luxe de détails et parce que ces résultats ne sont parfois que des approximations très raisonnables sans doute pour un spécialiste de ce domaine mais loin d'être évidentes pour un néophyte. On propose ici un éclairage essentiellement de traiteur de signal.

9.1 Hypothèses de base et notations

On note $x_1(n)$ et $x_2(n)$ les deux canaux stéréo. Pour pouvoir faire le lien avec des résultats standards de traitement du signal, il est commode de pouvoir interpréter ces deux signaux comme étant la réalisation de p.a. stationnaires, ergodiques, centrés $X_1(n)$ et $X_2(n)$ de puissances $\mathbb{E}\{X_1^2(n)\} = \sigma_{X_1}^2$ et $\mathbb{E}\{X_2^2(n)\} = \sigma_{X_2}^2$ et de fonction d'intercovariance normalisée

$$\rho(n_0) = \frac{\mathbb{E}\{X_1(n)X_2(n+n_0)\}}{\sqrt{\sigma_{X_1}^2\sigma_{X_2}^2}}.$$

On notera

$$\Gamma = \mathbb{E}\{\begin{bmatrix} X_1(n) \\ X_2(n) \end{bmatrix}[X_1(n)X_2(n)]\} = \begin{bmatrix} \sigma_{X_1}^2 & \rho(0)\sigma_{X_1}\sigma_{X_2} \\ \rho(0)\sigma_{X_1}\sigma_{X_2} & \sigma_{X_2}^2 \end{bmatrix}$$

$$\Gamma = \sigma_{X_1}\ \sigma_{X_2}\Gamma' = \sigma_{X_1}\sigma_{X_2}\begin{bmatrix} c & \rho(0) \\ \rho(0) & 1/c \end{bmatrix} \tag{9.1}$$

la matrice d'intercovariance en posant $c = \sigma_{X_1}/\sigma_{X_2}$.

Les trois indices intercanaux présentés au chapitre précédent, la différence d'intensité ICLD, la différence de temps ICTD et la cohérence ICC, sont obtenus à partir d'une estimation des puissances et de la fonction d'intercovariance

$$\begin{aligned} ICLD &= 10\log_{10}\frac{\hat{\sigma}_{X_1}^2}{\hat{\sigma}_{X_2}^2} = 20\log_{10} c \\ ICTD &= \arg\max_{n_0} |\hat{\rho}(n_0)| \\ ICC &= |\hat{\rho}(n_0 = ICTD)|. \end{aligned} \tag{9.2}$$

Le rapport des puissances est exprimé en dB. Il prend typiquement ses valeurs dans l'intervalle [-40, +40] dB. L'ICTD est exprimé en nombre d'échantillons. L'ICC, compris entre 0 et 1, donne l'étendue de l'événement auditif. Proche de 1, l'événement auditif est perçu de façon relativement compact. Voisin de 0, on perçoit deux événements distincts. Approximativement égal à 0.4, il en résulte une sensation équilibrée d'étendue pour donner un ordre de grandeur.

En réalité des études de psychoacoustique montrent que le calcul de ces indices intercanaux doit être réalisé dans le plan temps/ fréquences avec une actualisation temporelle toutes les 30 à 50 ms et une résolution fréquentielle homogène aux échelles Bark. C'est l'hypothèse de base la plus fondamentale de ce problème. La décomposition temps/fréquences la plus adaptée semble donc être celle obtenue par un banc de filtres cochléaires. Notons que cette décomposition doit être réalisée aussi bien à l'émetteur qu'au récepteur. L'emploi d'un banc de filtres cochléaires n'étant pas envisageable pour des raisons de complexité de traitement et également à cause de la difficulté de définir le banc de filtres de synthèse assurant la reconstruction parfaite, il faut donc se rabattre sur l'emploi d'un banc de filtres ou d'une transformée plus simple, par exemple la transformée de Fourier discrète.

Pour ne pas alourdir les notations, on note $\underline{x} = [x(0) \cdots x(N-1)]^t$ le vecteur construit à partir du signal $x(n)$ observé dans la fenêtre d'analyse courante de durée N échantillons et $\underline{X} = A\underline{x}$ le vecteur transformé[1]. La transformation utilisée habituellement est la transformée de Fourier à court terme avec une fenêtre de pondération recouvrante de 50%. Pour simplifier le formalisme, on supposera que A est la matrice de Fourier[2] de dimension $N \times N$ vérifiant $A^t A^* = I$.

9.2 Détermination des indices intercanaux

9.2.1 Estimation de la puissance et de l'intercovariance

On exploite les estimateurs standards de la puissance d'un signal et de l'intercovariance entre deux signaux. On obtient pour la puissance

$$\hat{\sigma}_X^2 = \frac{1}{N}\sum_{n=0}^{N-1} x^2(n) = \frac{1}{N}\underline{x}^t\underline{x} = \frac{1}{N}\underline{X}^t\underline{X}^* = \frac{||\underline{X}||^2}{N}$$

et pour le coefficient $\hat{\rho}(0)$

$$\hat{\rho}(0) = \frac{\sum_{n=0}^{N-1} x_1(n)x_2(n)}{\sqrt{\sum_{n=0}^{N-1} x_1^2(n) \sum_{n=0}^{N-1} x_2^2(n)}} = \frac{\underline{X}_1^t\underline{X}_2^*}{||\underline{X}_1||\ ||\underline{X}_2||}.$$

En réalité on désire une estimation de la fonction d'intercovariance pour $n_0 \neq 0$. Il paraît commode d'introduire le vecteur $\underline{x}_3$ déduit de $\underline{x}_2$ par décalage circulaire (à gauche)

$$\underline{x}_3(n_0) = [x_2(n_0) \cdots x_2(N-1)x_2(0) \cdots x_2(n_0-1)]^t.$$

On obtient

$$\hat{\rho}(n_0) = \frac{\underline{x}_1^t\ \underline{x}_3(n_0)}{||\underline{x}_1||\ ||\underline{x}_3(n_0)||} = \frac{\underline{x}_1^t\ \underline{x}_3(n_0)}{||\underline{x}_1||\ ||\underline{x}_2||} = \frac{\underline{X}_1^t\ \underline{X}_3^*(n_0)}{||\underline{X}_1||\ ||\underline{X}_2||}.$$

Comme

$$X_3(k, n_0) = \sum_{n=0}^{N-1} x_2(n + n_0\ mod\ N)\exp(-j2\pi\frac{k}{N}n) = X_2(k)\exp(j2\pi\frac{k}{N}n_0)$$

[1] Conflit de notation : $X(n)$ spécifie un p.a. et $\underline{X}$ un vecteur transformé. Comme la référence au p.a. est peu fréquente, ce conflit n'est pas très gênant.

[2] La définition de la TFD entraîne une matrice non normalisée. On la supposera ici normalisée pour simplifier.

on a

$$\underline{X}_3^*(n_0) = diag[1 \ \exp(-j2\pi\frac{1}{N}n_0)\cdots\exp(-j2\pi\frac{N-1}{N}n_0)] \ \underline{X}_2^*. \tag{9.3}$$

Donnons l'expression du produit scalaire $\underline{X}_1^t\underline{X}_2^*$ en exploitant la propriété de symétrie hermitienne. On a

$$\underline{X}_1^t\underline{X}_2^* = X_1(0)X_2(0) + X_1(\frac{N}{2})X_2(\frac{N}{2}) + \sum_{k=1}^{N/2-1}[X_1(k)X_2^*(k) + X_1(N-k)X_2^*(N-k)]$$

$$\underline{X}_1^t\underline{X}_2^* = X_1(0)X_2(0) + X_1(\frac{N}{2})X_2(\frac{N}{2}) + \sum_{k=1}^{N/2-1}[X_1(k)X_2^*(k) + X_1^*(k)X_2(k)]$$

$$\underline{X}_1^t\underline{X}_2^* = X_1(0)X_2(0) + X_1(\frac{N}{2})X_2(\frac{N}{2}) + 2\Re\sum_{k=1}^{N/2-1}X_1(k)X_2^*(k)\}.$$

On montre, figure 9.1, à gauche le signal stéréophonique et à droite la fonction d'intercovariance correspondante. Pour ce signal stéréophonique, la différence d'intensité intercanale est

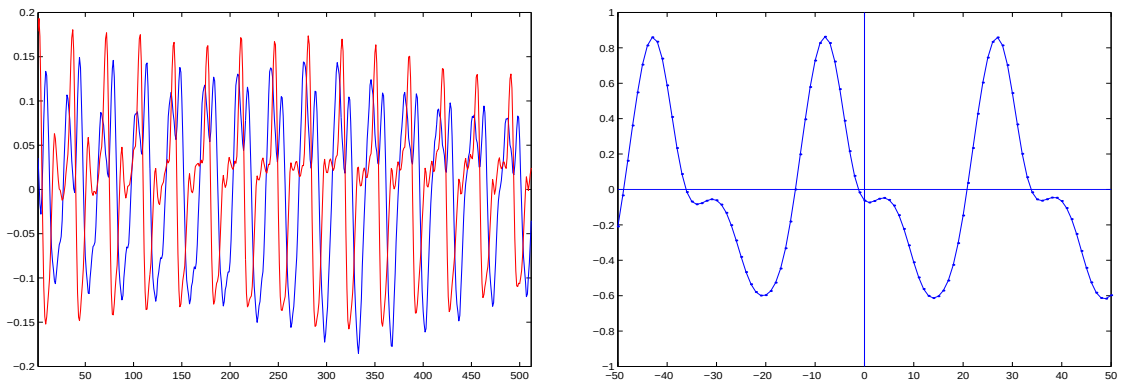

FIG. 9.1 – A gauche signal stéréo, à droite fonction d'intercovariance correspondante.

approximativement égale à 0 dB, la différence de temps est de 8 échantillons correspondant à 0.2 ms et la cohérence est légèrement supérieure à 0.8, caractéristiques d'une source sonore bien localisée approximativement à égale distance des deux haut-parleurs.

9.2.2 Evaluation des indices intercanaux

Les indices intercanaux sont calculés dans chaque "tuile" de la décomposition temps/fréquences. On note $\underline{X}(b)$ le sous-vecteur de $\underline{X}$ spécifique de la bande critique b

$$\underline{X}(b) = [X(k_b)\cdots X(k_{b+1}-1)]^t$$

où k_b spécifie la première composante de la b-ème bande critique. Il ne comporte que les composantes correspondant à des fréquences comprises entre 0 et $f_e/2$ mais il ne faut pas oublier de prendre en compte, lorsque c'est nécessaire, les composantes comprises entre $f_e/2$ et f_e. Ainsi un produit scalaire de la forme $\underline{X}_1^t\underline{X}_2^*$ qui fournit une valeur réelle doit s'écrire

$$\underline{X}_1^t(b)\underline{X}_2^*(b) + \underline{X}_2^t(b)\underline{X}_1^*(b) = 2\Re\{\underline{X}_1^t(b)\underline{X}_2^*(b)\}$$

dans le domaine des sous-bandes pour continuer à fournir une valeur réelle.

On obtient

– la différence d'intensité

$$ICLD(b) = 10\log_{10}\frac{\hat{\sigma}^2_{X_1}}{\hat{\sigma}^2_{X_2}} = 10\log_{10}\frac{\sum_{k=k_b}^{k_{b+1}-1}|X_1(k)|^2}{\sum_{k=k_b}^{k_{b+1}-1}|X_2(k)|^2} = 10\log_{10}\frac{||\underline{X}_1(b)||^2}{||\underline{X}_2(b)||^2}$$

– la différence de temps

$$ICTD(b) = \arg\max_{n_0\in\{-n_0^{max}\cdots n_0^{max}\}}|\hat{\rho}(n_0)|$$

avec

$$\hat{\rho}(n_0) = \frac{\Re\{\underline{X}_1^t(b)\ \underline{X}_3^*(b,n_0)\}}{||\underline{X}_1(b)||\ ||\underline{X}_2(b)||}$$

et n_0^{max} le plus petit entier tel que

$$\frac{k_b}{N}n_0^{max} \geq 1$$

– et la cohérence

$$ICC(b) = |\hat{\rho}(n_0 = ICTD(b))|.$$

Pour éviter une maximisation coûteuse en temps de calcul et des problèmes de précision délicats à traiter[3], le calcul de l'ICTD peut être remplacé par celui de la différence de phase intercanale

$$ICPD(b) = \arg\{\underline{X}_1^t(b)\ \underline{X}_2^*(b)\}$$

en se basant sur la justification suivante. L'exploitation de la cohérence est significative si elle prend une valeur proche de 1. C'est le cas si un signal est une version translatée de l'autre. Si on note $X(f)$ la transformée de Fourier d'un signal $x(n)$, la transformée de Fourier du signal $x(n+n_0)$ est égale à $X(f)\exp(j2\pi f n_0)$. Dans ce cas, l'ICTD et l'ICPD sont directement reliés puisque

$$\arg\{\underline{X}^t\ \underline{X}^*\exp(-j2\pi f n_0)\} = -2\pi f n_0$$

ce qui entraîne

$$ICPD = -2\pi f\ ICTD.$$

Connaissant l'ICPD, on en déduit l'ICC. En effet, comme

$$\underline{X}_3^*(b) = diag[\exp(-j2\pi\frac{k_b}{N}n_0)\cdots\exp(-j2\pi\frac{k_{b+1}-1}{N}n_0)]\ \underline{X}_2^*(b)$$

en conformité avec l'équation (9.3), on obtient l'approximation

$$\underline{X}_3^*(b) = \exp[-j\ ICPD(b)]\ \underline{X}_2^*(b).$$

On a donc

$$\underline{X}_1^t(b)\underline{X}_3^*(b) = \underline{X}_1^t(b)\underline{X}_2^*(b)\exp[-j\ ICPD(b)]$$

$$\underline{X}_1^t(b)\underline{X}_3^*(b) = |\underline{X}_1^t(b)\underline{X}_2^*(b)|\exp[j\ ICPD(b)]\exp[-j\ ICPD(b)] = |\underline{X}_1^t(b)\underline{X}_2^*(b)|$$

ce qui entraîne

$$ICC(b) = \frac{|\underline{X}_1^t(b)\ \underline{X}_2^*(b)|}{||\underline{X}_1(b)||\ ||\underline{X}_2(b)||}.$$

[3]Pour des bandes critiques correspondant à des fréquences élevées n_0^{max} est faible. Il faudrait alors s'intéresser à des valeurs de n_0 fractionnaires pour avoir une bonne précision.

9.2.3 Conclusion

Finalement, pour chaque tuile de la décomposition temps/fréquences, connaissant $\underline{X}_1(b)$ et $\underline{X}_2(b)$, on calcule les trois indices intercanaux

$$\begin{aligned} ICLD(b) &= 10\log_{10}\frac{||\underline{X}_1(b)||^2}{||\underline{X}_2(b)||^2} \\ ICPD(b) &= \arg\{\underline{X}_1^t(b)\ \underline{X}_2^*(b)\} \\ ICC(b) &= \frac{|\underline{X}_1^t(b)\ \underline{X}_2^*(b)|}{||\underline{X}_1(b)||\ ||\underline{X}_2(b)||}. \end{aligned} \tag{9.4}$$

9.3 Procédure de downmix

A l'émetteur, on dispose des deux canaux stéréo $x_1(n)$ et $x_2(n)$. On recherche le signal monophonique qui sera codé par la suite. La solution qui vient directement à l'esprit est de construire le signal $s(n) = [x_1(n) + x_2(n)]/\sqrt{2}$. Cette solution[4] n'est pas forcément la meilleure car elle ne garantit même pas la conservation de la puissance lorsque $\rho(0) \neq 0$

$$\sigma_S^2 = \frac{1}{2}\mathbb{E}\{[X_1(n) + X_2(n)]^2\} = \frac{1}{2}(\sigma_{X_1}^2 + \sigma_{X_2}^2 + 2\rho(0)\sigma_{X_1}\sigma_{X_2})$$

expression qui peut même être nulle si $\rho(0) = -1$. L'idée générale est de déterminer $s(n)$ par l'intermédiaire d'une rotation en écrivant

$$\begin{bmatrix} s(n) \\ s'(n) \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x_1(n) \\ x_2(n) \end{bmatrix} \tag{9.5}$$

et de déterminer l'angle θ qui maximise la puissance de $s(n)$. En réalité, le traitement que l'on va présenter pour commencer dans le domaine temporel sera réalisé dans le domaine fréquentiel, plus précisément pour chaque "tuile" de la décomposition temps/fréquences. Une synthèse OLA permettra de revenir ensuite au signal $s(n)$ dans le domaine temporel.

9.3.1 Développement dans le domaine temporel

Déterminons le vecteur (normalisé) $\underline{p} = [\cos(\theta)\ \sin(\theta)]^t$ maximisant la puissance de $S(n) = \underline{p}^t[X_1(n)X_2(n)]^t = \underline{p}^t\underline{X}(n)$. On obtient

$$\underline{p}^{opt} = \arg\max_{\underline{p}} \mathbb{E}\{[\underline{p}^t\underline{X}(n)]^2\} = \arg\max_{\underline{p}} \underline{p}^t\mathbb{E}\{\underline{X}(n)\underline{X}^t(n)\}\underline{p} = \arg\max_{\underline{p}} \underline{p}^t\Gamma\underline{p}$$

sous la contrainte $||\underline{p}|| = 1$. C'est un problème classique d'Analyse en Composantes Principales (ACP). On annule la dérivée de $\underline{p}^t\Gamma\underline{p} - \lambda[\underline{p}^t\underline{p} - 1]$ par rapport à $\underline{p}$. On obtient $\Gamma\underline{p} = \lambda\underline{p}$. Le paramètre λ est donc une valeur propre de la matrice Γ. En prémultipliant cette équation par $\underline{p}^t$, on obtient $\underline{p}^t\Gamma\underline{p} = \lambda\underline{p}^t\underline{p} = \lambda$. Le premier terme étant l'expression que l'on cherche à maximiser, il suffit de choisir le vecteur propre associé à la plus grande valeur propre de la matrice Γ.

Les valeurs propres de la matrice Γ' donnée par (9.1) ont pour expression

$$\lambda_{1,2} = \frac{1}{2}(c + 1/c \pm \sqrt{\Delta}) \quad \text{avec} \quad \Delta = (c - 1/c)^2 + 4\rho^2(0).$$

Les deux vecteurs propres étant orthogonaux, si on appelle $[\cos(\theta)\sin(\theta)]^t$ le premier vecteur propre (normalisé), le deuxième vecteur propre est nécessairement $[-\sin(\theta)\cos(\theta)]^t$. On a donc

$$\begin{bmatrix} c & \rho(0) \\ \rho(0) & 1/c \end{bmatrix} \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} = V\Lambda.$$

[4] appelée downmix passif dans la littérature

La plus grande valeur propre est

$$\lambda = \frac{1}{2}(c + 1/c + \sqrt{\Delta}).$$

Si on l'associe au premier vecteur propre on obtient

$$\tan(\theta) = \frac{\rho(0)}{\lambda - 1/c} = \frac{2\rho(0)}{c - 1/c + \sqrt{\Delta}}.$$

Si on l'associe au deuxième vecteur propre on obtient

$$\tan(\theta) = \frac{\rho(0)}{c - \lambda} = \frac{2\rho(0)}{c - 1/c - \sqrt{\Delta}}.$$

Dans les deux cas, si on exploite la relation

$$\tan(2\theta) = \frac{2\tan(\theta)}{1 - \tan^2(\theta)}$$

on obtient

$$\theta = \frac{1}{2}\arctan\left(\frac{2\rho(0)}{c - 1/c}\right)$$

angle compris entre $-\pi/4$ et $+\pi/4$. Lorsque θ est positif, c'est à dire quand $\rho(0)$ et $c - 1/c$ sont de même signe, alors

$$s(n) = \cos(\theta)x_1(n) + \sin(\theta)x_2(n).$$

Lorsque θ est négatif, on doit choisir

$$s(n) = -\sin(\theta)x_1(n) + \cos(\theta)x_2(n).$$

Il est plus simple de toujours calculer le signal dominant par la première relation en choisissant

$$\theta^{opt} = \mod\left[\frac{1}{2}\arctan\left(\frac{2\rho(0)}{c - 1/c}\right), \frac{\pi}{2}\right]. \tag{9.6}$$

Lorsque l'on choisit $\theta = \theta^{opt}$, la relation (9.5) est simplement la transformée de Karhunen Loeve appliquée au vecteur $\underline{X}(n)$

$$\begin{bmatrix} S(n) \\ S'(n) \end{bmatrix} = V^t \begin{bmatrix} X_1(n) \\ X_2(n) \end{bmatrix}$$

On a

$$\mathbb{E}\{\begin{bmatrix} S(n) \\ S'(n) \end{bmatrix}[S(n)S'(n)]\} = V^t\Gamma V = \sigma_{X_1}\sigma_{X_2}\Lambda.$$

Les deux signaux $S(n)$ et $S'(n)$ ont alors pour puissances respectives $\sigma_S^2 = \sigma_{X_1}\sigma_{X_2}\lambda_1$ et $\sigma_{S'}^2 = \sigma_{X_1}\sigma_{X_2}\lambda_2$ et ils ne sont pas corrélés. On peut remarquer que le rapport des deux valeurs propres c'est à dire des deux puissances est donné par

$$\frac{\lambda_2}{\lambda_1} = \frac{c + 1/c - \sqrt{\Delta}}{c + 1/c + \sqrt{\Delta}} = \frac{1 - \sqrt{\mu}}{1 + \sqrt{\mu}} \quad \text{avec} \quad \mu = 1 + \frac{4\rho^2(0) - 4}{(c + 1/c)^2}. \tag{9.7}$$

Pour déterminer θ^{opt}, une autre démarche est possible. En effet, à partir de (9.5), on obtient directement

$$\sigma_S^2 = \cos^2(\theta)\sigma_{X_1}^2 + 2\cos(\theta)\sin(\theta)\rho(0)\sigma_{X_1}\sigma_{X_2} + \sin^2(\theta)\sigma_{X_2}^2$$
$$\sigma_{S'}^2 = \sin^2(\theta)\sigma_{X_1}^2 - 2\cos(\theta)\sin(\theta)\rho(0)\sigma_{X_1}\sigma_{X_2} + \cos^2(\theta)\sigma_{X_2}^2$$

ou

$$\tilde{\sigma}_S^2 = \frac{\sigma_S^2}{\sigma_{X_1}\sigma_{X_2}} = c\cos^2(\theta) + 2\rho(0)\cos(\theta)\sin(\theta) + \frac{1}{c}\sin^2(\theta) = \frac{1}{2}(c+\frac{1}{c}) + \frac{y(\theta)}{2}$$

$$\tilde{\sigma}_{S'}^2 = \frac{\sigma_{S'}^2}{\sigma_{X_1}\sigma_{X_2}} = c\sin^2(\theta) - 2\rho(0)\cos(\theta)\sin(\theta) + \frac{1}{c}\cos^2(\theta) = \frac{1}{2}(c+\frac{1}{c}) - \frac{y(\theta)}{2}$$

avec

$$y(\theta) = (c - 1/c)\cos(2\theta) + 2\rho(0)\sin(2\theta).$$

Comme

$$\frac{\partial y(\theta)}{\partial \theta} = -2(c - 1/c)\sin(2\theta) + 4\rho(0)\cos(2\theta)$$

la fonction $y(\theta)$ présente un extremum pour

$$\theta = \frac{1}{2}\arctan\left(\frac{2\rho(0)}{c - 1/c}\right).$$

L'examen des 4 cas possibles suivant le signe de $\rho(0)$ et de $c - 1/c$ conduit à prendre θ modulo $\pi/2$ comme dans l'équation (9.6). Connaissant θ^{opt}, on en déduit $s_1(n)$ et $s_2(n)$.

9.3.2 Dans le domaine fréquentiel

Ce développement réalisé dans le domaine temporel peut être réalisé pour chaque "tuile" de la décomposition temps/fréquences. Il faut déterminer les angles $\theta^{opt}(b)$ pour $b = 1 \cdots B$. Ils dépendent uniquement de

$$c(b) = \frac{||\underline{X}_1(b)||}{||\underline{X}_2(b)||}$$

et de

$$\hat{\rho}(0,b) = \frac{1}{2}\frac{\underline{X}_1^t(b)\underline{X}_2^*(b) + \underline{X}_2^t(b)\underline{X}_1^*(b)}{||\underline{X}_1(b)||\ ||\underline{X}_2(b)||} = \frac{\Re\{\underline{X}_1^t(b)\underline{X}_2^*(b)\}}{||\underline{X}_1(b)||\ ||\underline{X}_2(b)||}$$

$$\hat{\rho}(0,b) = \frac{|\underline{X}_1^t(b)\underline{X}_2^*(b)|}{||\underline{X}_1(b)||\ ||\underline{X}_2(b)||}\cos[ICPD(b)]$$

$$\hat{\rho}(0,b) = ICC(b)\ \cos[ICPD(b)].$$

Les trois indices intercanaux donnés par (9.4) sont donc suffisants pour calculer le signal monophonique $s(n)$.

9.4 Au récepteur

On décode le train binaire transmis contenant les informations spécifiques du signal monophonique $s(n)$ et des trois indices binauraux. On conservera pour $s(n)$ et ces trois indices les mêmes notations par la suite bien qu'il y ait nécessairement une distortion due à l'étape de quantification. On supposera cette distorsion négligeable devant les approximations résultant de l'étape de modélisation. En particulier on supposera que la quantification de ces indices ne les dénature pas (ils gardent la même influence perceptuelle).

On a vu qu'à l'émetteur on caractérisait $s(n)$ par l'intermédiaire d'une rotation pour chaque tuile b de la décomposition temps/fréquences et pour chaque composante $k \in \{k_b \cdots k_{b+1} - 1\}$ de la b-ème bande critique en écrivant

$$\begin{bmatrix} S(k) \\ S'(k) \end{bmatrix} = \begin{bmatrix} \cos(\theta^{opt}(b)) & \sin(\theta^{opt}(b)) \\ -\sin(\theta^{opt}(b)) & \cos(\theta^{opt}(b)) \end{bmatrix} \begin{bmatrix} X_1(k) \\ X_2(k) \end{bmatrix}$$

ou pour simplifier les notations

$$\begin{bmatrix} S \\ S' \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}.$$

Au récepteur il faut faire les opérations inverses

$$\begin{bmatrix} \hat{X}_1 \\ \hat{X}_2 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} S \\ S' \end{bmatrix}.$$

Si on connaissait S' ou $s'(n)$, on obtiendrait[5] $\hat{X}_1 = X_1$ et $\hat{X}_2 = X_2$. Comme on ne le connaît pas, on introduit plusieurs modifications.

9.4.1 Génération de $s'(n)$

Il s'agit d'obtenir un signal qui puisse se substituer à $s'(n)$. Intéressons nous d'abord à la procédure de downmix et cherchons une interprétation physique dans le cas d'une restitution en stéréophonie. On supposera les deux signaux $x_1(n)$ et $x_2(n)$ alimentant deux hauts-parleurs placés aux sommets A et B d'un triangle isocèle caractérisé par l'angle BOx = xOA = ψ_0 comme le montre le schéma de la figure 9.2. Par convention, le signal $x_1(n)$ correspond au canal gauche et

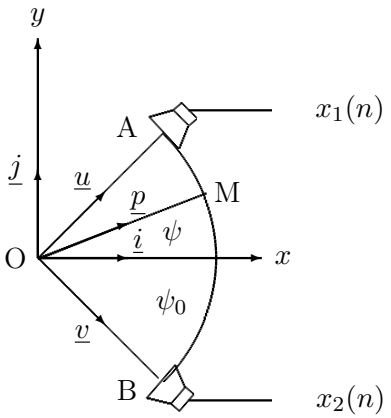

FIG. 9.2 – Restitution stéréophonique.

$x_2(n)$ au canal droit. Le troisième sommet O correspond à la place de l'auditeur ("sweet spot"). Les deux canaux créent une source virtuelle positionnée en M qui est caractérisée par l'angle xOM égal à ψ, l'azimuth de la source. En remarquant que $\psi = \psi_0$ lorsque c est très grand (la source virtuelle est en A) et $\psi = -\psi_0$ lorsque c est très petit (la source virtuelle est en B), on obtient

$$\psi = \psi_0(-\frac{4}{\pi}\theta + 1).$$

On montre, figure 9.3, l'angle ψ en fonction de c pour différentes valeurs de $\rho(0)$ lorsque $\psi_0 = 30$ degrés.

En première approximation, dans le cas d'une seule source sonore, on peut admettre que le signal $s(n)$ spécifie cette source virtuelle et que le signal $s'(n)$ représente l'information "orthogonale" à savoir l'effet dû à la réverbération. Comme on sait que la réverbération est composée d'un grand nombre de versions retardées et atténuées de la source on peut admettre que $s'(n)$ doit être une version décorrélée de $s(n)$ mais avoir la même enveloppe spectro-temporelle et que la façon la plus élémentaire est de prendre une simple version retardée ou de réaliser un filtrage

[5]Il faut rajouter des conditions dues à la synthèse OLA.

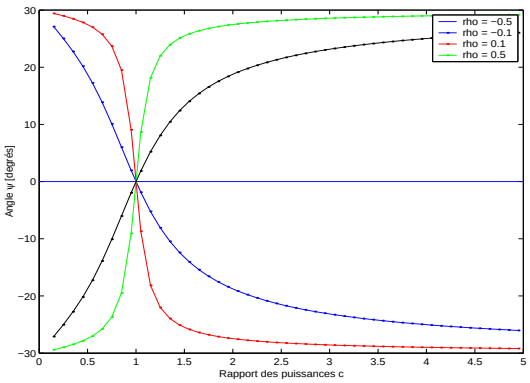

FIG. 9.3 – Angle ψ en fonction de c pour différentes valeurs de $\rho(0)$.

par un filtre en peigne et par un filtre passe-tout. Dans [50], Breebaart propose d'utiliser un filtre de réponse impulsionnelle

$$h(n) = \frac{2}{N}\sum_{k=0}^{N/2}\cos\left(2\pi\frac{k}{N}n + 2\pi\frac{k}{N}(k-1)\right)$$

avec $N = 640$. On montre que le spectre d'amplitude est plat et que le spectre de phase est très haché. On peut également vérifier qu'il a un gain en puissance égal à 1.

9.4.2 Réglage des puissances

On a vu que S et S' à l'émetteur n'ont pas du tout la même puissance. Si au récepteur S' est déduit de S, on supposera que l'on engendre d'abord une composante de même puissance. Il faudra donc ensuite régler convenablement les puissances. Si on écrit

$$\begin{bmatrix} \hat{S} \\ \hat{S}' \end{bmatrix} = \begin{bmatrix} \cos(\varphi) & 0 \\ 0 & \sin(\varphi) \end{bmatrix} \begin{bmatrix} S \\ S' \end{bmatrix}$$

on voit que cette paramétrisation entraîne la conservation des puissances $\sigma^2_{\hat{S}} + \sigma^2_{\hat{S}'} = \sigma^2_S$. Il suffit de s'assurer ensuite que le rapport des puissances soit égal au rapport des valeurs propres donné par (9.7)

$$\tan(\varphi) = \frac{\lambda_2}{\lambda_1} = \frac{1-\sqrt{\mu}}{1+\sqrt{\mu}} \quad \text{avec} \quad \mu = 1 + \frac{4\rho^2(0)-4}{(c+1/c)^2}.$$

9.4.3 Alignement de phase

On introduit un "alignement de phase" comme le propose Breebaart *et al.* [50] en rajoutant une matrice supplémentaire sous la forme

$$\begin{bmatrix} \hat{X}_1 \\ \hat{X}_2 \end{bmatrix} = \begin{bmatrix} \exp(j\Phi_1) & 0 \\ 0 & \exp(j\Phi_2) \end{bmatrix} \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} \cos(\varphi) & 0 \\ 0 & \sin(\varphi) \end{bmatrix} \begin{bmatrix} S \\ S' \end{bmatrix}.$$

Donnons une interprétation de Φ_1 et Φ_2 dans le plan complexe. On remarquera d'abord qu'une interprétation dans le plan complexe ne peut se faire qu'avec des scalaires (complexes) et non des vecteurs alors que l'on veut interpréter le comportement d'un certain nombre de composantes dans une sous bande. En réalité on interprétera un comportement moyen spécifique de chaque sous bande.

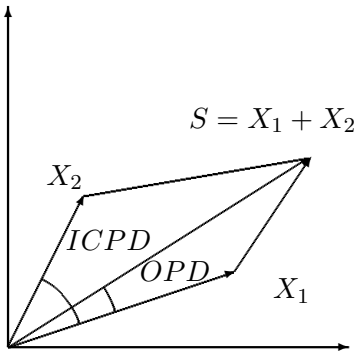

FIG. 9.4 – Dans le plan complexe : cas où $S = X_1 + X_2$.

Plaçons nous à l'émetteur et supposons donc que les vecteurs $\underline{X}_1(b)$ et $\underline{X}_2(b)$ puissent être représentés par deux nombres complexes X_1 et X_2. Considérons la représentation donnée à la figure 9.4. En appelant M_1 et M_2 les images des deux nombres complexes X_1 et X_2, la différence de phase intercanale

$$ICPD(b) = \arg\{\underline{X}_1^t(b)\ \underline{X}_2^*(b)\}$$

s'interprète comme l'angle $(Ox, OM_1) - (Ox, OM_2) = -(OM_1, OM_2)$. On introduit la différence de phase totale (Overall Phase Difference)

$$OPD(b) = \arg\{\underline{X}_1^t(b)\ \underline{S}^*(b)\}.$$

dont l'interprétation dans le plan complexe est donnée à la figure 9.4. On voit que si on pose $\Phi_1 = OPD$ et $\Phi_2 = OPD - ICPD$, on obtiendra $\hat{X}_1$ et $\hat{X}_2$ au récepteur aligné "en moyenne" avec X_1 et X_2 à l'émetteur.

Connaissant $\hat{X}_1$ et $\hat{X}_2$, deux synthèses OLA, une pour chaque voie, permettent ensuite de calculer $\hat{x}_1(n)$ et $\hat{x}_2(n)$.

9.4.4 Informations à transmettre dans le canal

Pour chaque tuile de la décomposition temps/fréquences on doit transmettre l'ICLD, l'ICPD, l'ICC et *a priori* l'OPD. En réalité le transfert de l'OPD n'est pas nécessaire.

En effet, si on réalise un downmix passif $S = X_1 + X_2$, Lapierre et Lefebvre [52] ont montré que

$$OPD = \arg\{\underline{X}_1^t\ (\underline{X}_1^* + \underline{X}_2^*)\} = \arg\{||\underline{X}_1||^2 + |\underline{X}_1^t\ \underline{X_2}^*| \exp(j\ ICPD)\}$$

$$OPD = \arg\{\frac{||\underline{X}_1||}{||\underline{X}_2||} + \frac{|\underline{X}_1^t\ \underline{X_2}^*|}{||\underline{X}_1||\ ||\underline{X}_2||} \exp(j\ ICPD)\}$$

$$OPD = \arg\{c + ICC \exp(j\ ICPD)\}.$$

Ce résultat se généralise directement dans le cas d'un downmix actif. En effet

$$OPD = \arg\{\underline{X}_1^t\ [\cos(\theta)\underline{X}_1^* + \sin(\theta)\underline{X}_2^*]\}$$

$$OPD = \arg\{\cos(\theta)\frac{||\underline{X}_1||}{||\underline{X}_2||} + \sin(\theta)\frac{|\underline{X}_1^t\ \underline{X_2}^*|}{||\underline{X}_1||\ ||\underline{X}_2||} \exp(jICPD)\}$$

$$OPD = \arg\{\cos(\theta)\ c + \sin(\theta)\ ICC \exp(j\ ICPD)\}.$$

9.5 Draft International Standard

Le document [30] montre qu'en réalité il n'y a pas de downmix actif. Le signal monophonique est directement donné par $s(n) = [x_1(n) + x_2(n)]/2$.

On donne, table 9.1, les dictionnaires de quantification pour les paramètres ICLD, ICPD et ICC. Apparemment l'OPD est également quantifié avec le dictionnaire de l'ICPD et transmis. Dans l'hypothèse où on ne transfère que les 3 premiers paramètres, cela coûte 10 bits par fenêtre

ICLD	-25	-18	-14	-10	-7	-4	-2	0	2
	4	7	10	14	18	25			
ICPD	0	$\pi/4$	$\pi/2$	$3\pi/4$	π	$5\pi/4$	$3\pi/2$	$7\pi/4$	2π
ICC	1	0.937	0.841	0.6	0.367	0	-0.589	-1	

TAB. 9.1 – Dictionnaire de quantification pour l'ICLD, l'ICPD et l'OPD.

d'analyse. Si on transfère ces paramètres toutes les 20 ms, le débit correspondant est de 2 kbit/s.

On montre, figure 9.5, les angles θ calculés au récepteur en fonction du temps pour l'ensemble des bandes critiques lorsque le signal stéréophonique traité est le résultat d'un double panoramique d'intensité. On rappelle que pour un panoramique d'intensité, les deux canaux $x_1(n)$ et $x_2(n)$ sont créés à partir d'une source unique $s(n)$ de la façon suivante $x_1(n) = g_1 \times s(n)$ et $x_2(n) = g_2 \times s(n)$. Ici on a simultanément un violon paraissant se déplacer du haut parleur gauche vers le haut parleur droit puis retour vers le haut parleur gauche et un piano paraissant faire le chemin inverse.

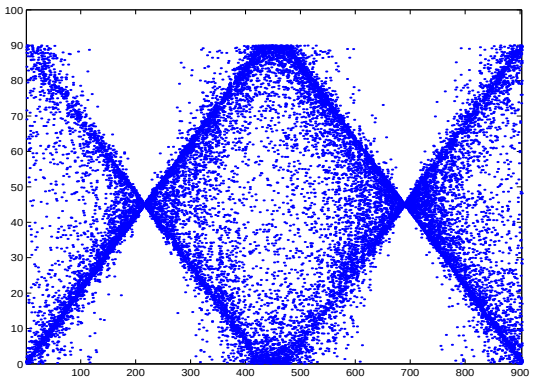

FIG. 9.5 – Angles θ en fonction du temps calculés pour l'ensemble des bandes critiques.

Troisième partie

Programmes matlab

Chapitre 10

Un codeur de parole

10.1 Introduction

Le programme proposé ne simule pas à proprement parler un codeur de parole. A partir d'un fichier de signal de parole en bande téléphonique ($F_e = 8$ kHz) au format wave (échantillons sur 16 bits), il ne construit pas un fichier contenant les mots de code à transmettre (le bitstream). Il se contente de l'étape de modélisation consistant à déterminer les paramètres du filtre de synthèse et les paramètres caractérisant le signal d'entrée de ce filtre chaque fenêtre d'analyse. Pour simuler un vrai codeur, il reste à réaliser l'étape de quantification de ces paramètres et à séparer clairement ce que doivent réaliser le codeur et le décodeur. Comme un codeur de parole obéit à un schéma d'analyse par la synthèse, le programme de simulation du décodeur se réduit à un petit sous-ensemble du programme complet (plus le décodage des mots de code).

Avec les valeurs proposés pour la longueur de la fenêtre d'analyse, l'ordre du filtre de synthèse, la dimension des dictionnaires, ce programme simule un codeur à un débit de l'ordre de 10 kbit/s.

10.2 Script de la fonction appelante

```
clear

% Paramètres
Fe = 8000;
N = 320;
M = 160;
P = 12;
Nbre_sfen = 4;
M_prime = M/Nbre_sfen;
L_plt = 256;
L = 512;
Dico_C = randn(M/Nbre_sfen,L);
Nbre_vecteurs = 2;
Gamma = 0.8;

% Estimation du débit
nbre_bits_a = 30;
nbre_bits_gain = 5;
D = (nbre_bits_a + Nbre_sfen*Nbre_vecteurs*(nbre_bits_gain+log2(L)))*Fe/M;
D = D + Nbre_sfen*(nbre_bits_gain+log2(L_plt))*Fe/M;
fprintf('Débit = %d kbits/s\n', round(D/1000))
```

```
fprintf('Gamma = %.2f\n', Gamma)

% Lecture du signal
prefixe = 'C:\Documents and Settings\Nicolas\Mes documents';
prefixe_in  = [prefixe '\mes_signaux\signaux_ref\fe_8\'];
prefixe_out = [prefixe '\mes_signaux\signaux_temp\'];
signal_xn = wavread([prefixe_in 'voix_femme_1.wav'], [0.15 3]*Fe);
Nbre_ech = length(signal_xn);

Nbre_ech_visu = Fe/4;
axe_temps = (0:Nbre_ech_visu-1)'/Fe;
vmax_xn = max(abs(signal_xn));
figure(1); hold off
plot(axe_temps, signal_xn(1:Nbre_ech_visu)); hold on
axis([0 Nbre_ech_visu/Fe 1.1*max(abs(signal_xn))*[-1 1]])

% Quelques réservations
etat_yn = zeros(P,1);
etat_pn = zeros(P,1);
etat_ringing = zeros(P,1);
etat_xn_hat = zeros(P,1);
signal_yn = zeros(Nbre_ech, 1);
signal_yn_hat = zeros(Nbre_ech, 1);
signal_xn_hat = zeros(Nbre_ech, 1);
buffer_yn_hat = zeros(L_plt+M_prime,1);
vn = 0.56 - 0.44*cos(2*pi*(0:N-1)'/(N-1));

% Traitement par fenetre
Nbre_fen = fix((Nbre_ech-N)/M) + 1;
Nbre_fen_visu = fix((Nbre_ech_visu-N)/M) + 1;
for no_fen = 1:Nbre_fen
   n1 = (no_fen-1)*M + 1;
   n2 = n1 + N - 1;
   n3 = n1 + (N-M)/2;
   n4 = n3 + M - 1;

   % Analyse LPC et calcul du signal résiduel
   xn = vn.*signal_xn(n1:n2);
   [a sigma2] = analyse_lpc(xn, P);
   xn = signal_xn(n3:n4);
   [yn, etat_yn] = filtre_ma(a, xn, etat_yn);
   signal_yn(n3:n4) = yn;

   % Calcul du signal perceptuel
   a_percep = a.*(Gamma.^(0:P)');
   [pn, etat_pn] = filtre_ar(1, a_percep, yn, etat_pn);

   % Filtrage du dictionnaire "stochastique"
   Dico_F = zeros(size(Dico_C));
   alpha = zeros(L,1);
```

```
for i = 1:L
    Dico_F(:,i) = filter(1, a_percep, Dico_C(:,i));
    alpha(i) = norm(Dico_F(:,i));
end

% Détermination de l'entrée du filtre de synthèse
yn_hat = zeros(M,1);
for no_sfen = 1:Nbre_sfen
    n5 = (no_sfen-1)*M_prime + 1;
    n6 = no_sfen*M_prime;

    % Construction de Dico_C_plt à partir de buffer_yn_hat
    Dico_C_plt = zeros(M_prime, L_plt);
    for i = 1:L_plt
        Dico_C_plt(:,i) = buffer_yn_hat(i:i+M_prime-1);
    end
    % Filtrage du dictionaire
    Dico_F_plt = zeros(M_prime,L_plt);
    alpha_plt = zeros(L_plt,1);
    for i = 1:L_plt
        Dico_F_plt(:,i) = filter(1, a_percep, Dico_C_plt(:,i));
        alpha_plt(i) = norm(Dico_F_plt(:,i));
    end

    % Recherche du 1er vecteur dans le dictionnaire adaptatif
    ringing = filtre_ar(1, a_percep, zeros(M_prime,1), etat_ringing);
    en = pn(n5:n6) - ringing;
    [gain_plt, indice_plt] = determination_gains_formes(en, Dico_F_plt, ...
        alpha_plt, 1);
    yn_hat(n5:n6) = yn_hat(n5:n6) + gain_plt*Dico_C_plt(:,indice_plt);
    en = en - gain_plt*Dico_F_plt(:,indice_plt);

    % Recherche dans le dictionnaire stochastique
    [gains, indices] = determination_gains_formes(en, Dico_F, alpha, ...
        Nbre_vecteurs);
    for k = 1:Nbre_vecteurs
        yn_hat(n5:n6) = yn_hat(n5:n6) + gains(k)*Dico_C(:,indices(k));
    end

    % Actualisation du buffer sauvegardant yn_hat
    for l = 1:L_plt
        buffer_yn_hat(l) = buffer_yn_hat(l+M_prime);
    end
    buffer_yn_hat(L_plt+1:L_plt+M_prime) = yn_hat(n5:n6);

    % Actualisation du ringing
    [ans, etat_ringing] = filtre_ar(1, a_percep, yn_hat(n5:n6), etat_ringing);
end
signal_yn_hat(n3:n4) = yn_hat;
```

```
% Calcul de xn_hat
[xn_hat, etat_xn_hat] = filtre_ar(1, a, yn_hat, etat_xn_hat);
signal_xn_hat(n3:n4) = xn_hat;

% Visualisations
if no_fen <= Nbre_fen_visu
    figure(1);
    plot(axe_temps(n1:n2), vmax_xn*vn, 'r');
    plot(axe_temps(1:n4), signal_xn_hat(1:n4), 'r');
    plot(axe_temps(n3)*[1 1], 1.1*vmax_xn*[-1 1], 'g')
    plot(axe_temps(n4)*[1 1], 1.1*vmax_xn*[-1 1], 'g')

    figure(3); hold off
    plot(xn); hold on
    plot(xn_hat, 'r')
    plot(xn-xn_hat, 'k')
    plot([1 M], [0 0])
    for no_sfen = 1:Nbre_sfen
        n5 = (no_sfen-1)*M_prime + 1;
        plot(n5*[1 1], 1.1*max(abs(xn))*[-1 1], 'g')
    end
    axis([1 M 1.1*max(abs(xn))*[-1 1]])

    % Visualisations des spectres dans la fenetre d'analyse précédente
    off_set = 90;
    axe_freq = (0:N/2-1)*Fe/N/1000;
    if no_fen > 1
        n1 = (no_fen-2)*M + 1;
        n2 = n1 + N - 1;

        signal_pond = vn.*signal_xn(n1:n2);
        temp = (abs(fft(signal_pond)).^2)/N;
        perio_xn_db = 10*log10(temp(1:N/2)) + off_set;
        [a_bis sigma2_bis] = analyse_lpc(signal_pond, P);
        temp = abs(fft(a_bis,N)).^2;
        spectre_lpc_xn_db = 10*log10(sigma2_bis./temp(1:N/2)) + off_set;

        signal_pond = vn.*signal_xn_hat(n1:n2);
        temp = (abs(fft(signal_pond)).^2)/N;
        perio_xn_hat_db = 10*log10(temp(1:N/2)) + off_set;
        [a_bis sigma2_bis] = analyse_lpc(signal_pond, P);
        temp = abs(fft(a_bis,N)).^2;
        spectre_lpc_xn_hat_db = 10*log10(sigma2_bis./temp(1:N/2)) + off_set;

        signal_pond = vn.*(signal_xn(n1:n2)-signal_xn_hat(n1:n2));
        temp = (abs(fft(signal_pond)).^2)/N;
        perio_en_db = 10*log10(temp(1:N/2)) + off_set;

        figure(2); hold off
        plot(axe_freq, perio_xn_db, 'linewidth', 2); hold on
```

```
            plot(axe_freq, perio_xn_hat_db, 'r');
            plot(axe_freq, perio_en_db, 'k');
            axis([0 Fe/2000 0 100])
        end
        drawnow
        fprintf('pause \n'); pause
    else
        if rem(no_fen,20) ==0
            fprintf('no_fen = %3d/%3d\n', no_fen, Nbre_fen)
        end
    end
end

wavwrite(signal_xn, Fe, [prefixe_out 'voix_originale']);
wavwrite(signal_xn_hat, Fe, [prefixe_out 'codeur_celp_avec_plt_gamma']);
```

10.3 Script des fonctions appelées

```
function [a, sigma2] = analyse_lpc(xn, P)
N = length(xn);
rk = zeros(P+1,1);
for k = 1:P+1
   rk(k) = xn(1:N-k+1)'*xn(k:N)/N;
end
if rk(1) > 0
   R = toeplitz(rk(1:P));
   a = [1; -inv(R)*rk(2:P+1)];
   sigma2 = a'*rk;
else
   a = [1; zeros(P,1)];
   sigma2 = 0;
end

function [yn, etat] = filtre_ma(bi, xn, etat)
P = length(etat);
yn = zeros(size(xn));
for n = 1:length(xn)
   yn(n) = bi(1)*xn(n) + bi(2:P+1)'*etat;
   etat(2:P) = etat(1:P-1);
   etat(1) = xn(n);
end

function [yn, etat] = filtre_ar(b0, ai, xn, etat)
P = length(etat);
yn = zeros(size(xn));
for n = 1:length(xn)
   yn(n) = b0*xn(n) - ai(2:P+1)'*etat;
   etat(2:P) = etat(1:P-1);
   etat(1) = yn(n);
end
```

```
function [gains, indices] = determination_gains_formes(en, Dico, ...
   alpha, Nbre_vecteurs)
[M, L] = size(Dico);
gains = zeros(Nbre_vecteurs,1);
indices = ones(Nbre_vecteurs,1);
for k = 1:Nbre_vecteurs
   critere_max = 0;
   for j = 1:L
       beta = Dico(:,j)'*en;
       if alpha(j) ~= 0
           critere = abs(beta)/alpha(j);
       else
           critere = 0;
       end
       if critere > critere_max
           gains(k) = beta/(alpha(j)^2);
           indices(k) = j;
           critere_max = critere;
       end
   end
   en = en - gains(k)*Dico(:,indices(k));
end
```

Chapitre 11

Un codeur de musique

11.1 Introduction

Le programme proposé simule le codeur MPEG-1 Layer-1 comprimant un signal de musique échantillonné à 44.1 kHz à un débit compris entre 64 et 96 kbit/s. Seules les tables de quantification spécifiques de cette fréquence d'échantillonnage et de cette gamme de débit sont fournies. Contrairement au programme simulant un codeur de parole, celui-ci construit véritablement un fichier intermédiaire où sont rangés les mots de code. Toutefois la syntaxe spécifique du document normatif n'est pas respectée.

11.2 Script de la fonction appelante

```
clear all
global Fe N_mpeg M_mpeg D_mpeg

Fe = 44100;
N_mpeg = 512;
M_mpeg = 32;
D_mpeg = 12;
Debit_kbs = 64;

% Lecture du signal
prefixe = 'C:\Documents and Settings\Nicolas\Mes documents';
prefixe_in  = [prefixe '\mes_signaux\signaux_ref\fe_44.1\'];
prefixe_out = [prefixe '\mes_signaux\signaux_temp\'];
nom = 'partita_no3';
signal_xn = wavread([prefixe_in nom], [1.7 2.7]*Fe);
Nbre_ech = length(signal_xn);
signal_xn = (signal_xn(:,1)+signal_xn(:,2))/2;

figure(1); hold off
plot((1:Nbre_ech)/Fe, signal_xn); hold on
xlabel('Temps [s]')
ylabel ('Amplitudes')
axis([0 Nbre_ech/Fe 1.1*max(abs(signal_xn))*[-1 1]]);
drawnow

% Quelques initialisations
```

```
[hn, Mat_H] = init_banc_pqmf;
tables_quantif;
init_mpa1;

% Encodage
fprintf('Encodage à %d kbits/s du fichier %s \n', Debit_kbs, nom)
fid_bit_stream = fopen('bit_stream', 'w');
xn  = zeros(N_mpeg, 1);
xn_mpa = zeros(N_mpeg, 1);
Mat_sb = zeros(D_mpeg, M_mpeg);
hann = sqrt(8/3)/2*(ones(N_mpeg, 1) - cos(2*pi*(0:N_mpeg-1)'/N_mpeg));
Retard_yn  = N_mpeg + (D_mpeg-1)*M_mpeg;
Retard_mpa = N_mpeg + (D_mpeg-1)*M_mpeg/2;
no_ech_sb = 1;
Nbre_blocs = fix(Nbre_ech/M_mpeg);
for no_bloc = 1:Nbre_blocs
   % Actualisation de xn et de xn_mpa
   n1 = (no_bloc-1)*M_mpeg + 1;
   n2 = n1 + M_mpeg - 1;
   xn(1:N_mpeg-M_mpeg) = xn(M_mpeg+1:N_mpeg);
   xn(N_mpeg-M_mpeg+1:N_mpeg) = signal_xn(n1:n2);
   xn_mpa(1:N_mpeg-M_mpeg) = xn_mpa(M_mpeg+1:N_mpeg);
   xn_mpa(N_mpeg-M_mpeg+1:N_mpeg) = xn(2*N_mpeg-Retard_mpa-M_mpeg+1:2*N_mpeg-Retard_mpa);

   % Banc de filtres d'analyse
   Mat_sb(no_ech_sb,:) = (Mat_H*xn)';
   no_ech_sb = rem(no_bloc,D_mpeg) + 1;

   % Tous les 12 échantillons de sous-bande
   if rem(no_bloc, D_mpeg) == 0
       % Calcul du seuil de masquage
       [seuil1_db, seuil2_db, SMR_db_sb] = mpeg_mpa1(xn_mpa);
       % Allocation de bits
       NOQ = mpeg_alloc(SMR_db_sb, Debit_kbs);
       % Quantification et construction du bit-stream
       ecrit_bit_stream(fid_bit_stream, NOQ, Mat_sb)

       % Quelques visualisations
       off_set = 90;
       figure(2); hold off
       temp = fft(xn_mpa.*hann);
       X1 = 10*log10((abs(temp(1:N_mpeg/2)).^2)/N_mpeg);
       axe_freq = (0:N_mpeg/2-1)*Fe/1000/N_mpeg;
       plot(axe_freq, X1+off_set); hold on
       plot(axe_freq, seuil1_db+off_set, 'r');
       plot(axe_freq, seuil2_db+off_set, 'g');
       val_old = SMR_db_sb(1);
       for k = 1:M_mpeg
           ind = (k-1)*8+1:k*8+1;
           plot(ind(1)*Fe/1000/N_mpeg*[1 1], [val_old SMR_db_sb(k)])
```

```
            plot(ind*Fe/1000/N_mpeg, SMR_db_sb(k)*ones(9,1))
            val_old = SMR_db_sb(k);
        end
        plot([0 Fe/2], [0 0], ':');
        xlabel('Frequences [kHz]');
        ylabel('Puissances [dB]');
        axis([0 Fe/1000/2 -20 100]);
        drawnow
    end
    if rem(no_bloc,100) == 0
        fprintf(' %4.2f', no_bloc*M_mpeg/Fe)
        if rem(no_bloc,1000) == 0
            fprintf('\n')
        end
    end
end
fprintf('\n')
fclose(fid_bit_stream);

% Décodage
fprintf('Décodage \n')
fid_bit_stream = fopen('bit_stream');
signal_xn_hat = zeros(Nbre_ech,1);
xn_hat = zeros(N_mpeg,1);
Mat_sb_q = zeros(D_mpeg,M_mpeg);
Nbre_blocs_deb_transf = N_mpeg/M_mpeg + D_mpeg;
no_bloc = 0;
no_ech_sb = 1;
while 1
    no_bloc = no_bloc + 1;

    % Banc de filtres de synthèse
    xn_hat = xn_hat + Mat_H'*Mat_sb_q(no_ech_sb,:)';
    no_ech_sb = rem(no_bloc,D_mpeg) + 1;

    if no_bloc >= Nbre_blocs_deb_transf
        n1 = (no_bloc-Nbre_blocs_deb_transf)*M_mpeg + 1;
        n2 = n1 + M_mpeg - 1;
        signal_xn_hat(n1:n2) = xn_hat(1:M_mpeg);
    end
    xn_hat(1:N_mpeg-M_mpeg) = xn_hat(M_mpeg+1:N_mpeg);
    xn_hat(N_mpeg-M_mpeg+1:N_mpeg) = zeros(M_mpeg,1);

    if rem(no_bloc,D_mpeg) == 0
        [Mat_sb_q, stop] = lit_bit_stream(fid_bit_stream);
        if stop
            fprintf('\n')
            break
        end
    end
```

```
    if rem(no_bloc,100) == 0
        fprintf(' %4.2f', no_bloc*M_mpeg/Fe)
        if rem(no_bloc,1000) == 0
            fprintf('\n')
        end
    end
end
fclose(fid_bit_stream);

wavwrite(signal_xn, Fe, [prefixe_out 'signal_original']);
wavwrite(signal_xn_hat, Fe, [prefixe_out 'signal_reconstruit']);
wavwrite((signal_xn-signal_xn_hat(1:Nbre_ech)), ...
    Fe, [prefixe_out 'diff']);

figure(1);
plot((1:Nbre_ech)/Fe, signal_xn-signal_xn_hat(1:Nbre_ech), 'r');
```

11.3 Script des fonctions appelées

```
function [hn, Mat_H] = init_banc_pqmf

% Définition de la réponse impulsionnelle du filtre prototype
% et de la matrice de modulation du banc de filtres PQMF

global N_mpeg M_mpeg

temp = [ 0.202407  0.202283  0.201916  0.201307 ...
         0.200452  0.199359  0.198029  0.196465 ...
         0.194668  0.192648  0.190409  0.187952 ...
         0.185290  0.182422  0.179361  0.176113 ...
         0.172685  0.169084  0.165321  0.161407 ...
         0.157347  0.153153  0.148837  0.144402 ...
         0.139868  0.135239  0.130527  0.125745 ...
         0.120900  0.116004  0.111068  0.106105 ...
         0.101123  0.096135  0.091148  0.086174 ...
         0.081224  0.076307  0.071432  0.066610 ...
         0.061849  0.057155  0.052540  0.048011 ...
         0.043576  0.039242  0.035012  0.030899 ...
         0.026907  0.023036  0.019297  0.015693 ...
         0.012227  0.008901  0.005724  0.002692 ...
        -0.000189 -0.002919 -0.005495 -0.007917 ...
        -0.010185 -0.012303 -0.014264 -0.016074 ...
        -0.017733 -0.019243 -0.020608 -0.021827 ...
        -0.022906 -0.023845 -0.024652 -0.025326 ...
        -0.025873 -0.026300 -0.026604 -0.026799 ...
        -0.026882 -0.026863 -0.026747 -0.026537 ...
        -0.026238 -0.025855 -0.025399 -0.024867 ...
        -0.024271 -0.023616 -0.022904 -0.022143 ...
        -0.021336 -0.020492 -0.019613 -0.018706 ...
        -0.017773 -0.016824 -0.015858 -0.014882 ...
```

```
        -0.013900 -0.012915 -0.011936 -0.010960 ...
        -0.009994 -0.009039 -0.008103 -0.007183 ...
        -0.006285 -0.005411 -0.004564 -0.003744 ...
        -0.002954 -0.002196 -0.001470 -0.000777 ...
        -0.000121  0.000499  0.001084  0.001632 ...
         0.002142  0.002616  0.003051  0.003453 ...
         0.003814  0.004141  0.004435  0.004691 ...
         0.004915  0.005106  0.005265  0.005395 ...
         0.005495  0.005565  0.005611  0.005629 ...
         0.005624  0.005597  0.005549  0.005481 ...
         0.005397  0.005292  0.005176  0.005044 ...
         0.004901  0.004745  0.004580  0.004408 ...
         0.004227  0.004041  0.003852  0.003658 ...
         0.003461  0.003264  0.003067  0.002870 ...
         0.002673  0.002479  0.002287  0.002101 ...
         0.001918  0.001740  0.001567  0.001400 ...
         0.001238  0.001082  0.000936  0.000793 ...
         0.000658  0.000531  0.000413  0.000299 ...
         0.000194  0.000097  0.000005 -0.000078 ...
        -0.000154 -0.000224 -0.000286 -0.000343 ...
        -0.000394 -0.000440 -0.000477 -0.000510 ...
        -0.000539 -0.000561 -0.000580 -0.000596 ...
        -0.000604 -0.000612 -0.000615 -0.000615 ...
        -0.000612 -0.000607 -0.000599 -0.000588 ...
        -0.000575 -0.000561 -0.000545 -0.000529 ...
        -0.000513 -0.000494 -0.000475 -0.000456 ...
        -0.000434 -0.000415 -0.000397 -0.000375 ...
        -0.000356 -0.000337 -0.000316 -0.000299 ...
        -0.000281 -0.000262 -0.000245 -0.000229 ...
        -0.000213 -0.000197 -0.000183 -0.000170 ...
        -0.000156 -0.000143 -0.000132 -0.000121 ...
        -0.000111 -0.000103 -0.000094 -0.000084 ...
        -0.000078 -0.000070 -0.000065 -0.000057 ...
        -0.000051 -0.000046 -0.000043 -0.000038 ...
        -0.000035 -0.000030 -0.000027 -0.000024 ...
        -0.000022 -0.000019 -0.000019 -0.000016 ...
        -0.000013 -0.000013 -0.000011 -0.000011 ...
        -0.000008 -0.000008 -0.000005 -0.000005 ...
        -0.000005 -0.000005 -0.000003 -0.000003 ...
        -0.000003 -0.000003 -0.000003 -0.000003];
hn = [0 fliplr(temp(2:256)) temp]';

Mat_H = zeros(M_mpeg, N_mpeg);
for k = 0:M_mpeg-1
    t2 = ((2*k+1)*pi/(2*M_mpeg))*((0:N_mpeg-1)+M_mpeg/2);
    Mat_H(k+1,:) = hn'.*cos(t2);
end

function init_tables_quantif
```

```
% Initialisation de toutes les tables de quantification

global Fact_ech Rsb_iso_1_3 Rsb_iso_4_11 Rsb_iso_12_23 Rsb_iso_24_27

% Facteurs d'échelle
Fact_ech = [2.00000000000000, 1.58740105196820, 1.25992104989487, ...
            1.00000000000000, 0.79370052598410, 0.62996052494744, ...
            0.50000000000000, 0.39685026299205, 0.31498026247372, ...
            0.25000000000000, 0.19842513149602, 0.15749013123686, ...
            0.12500000000000, 0.09921256574801, 0.07874506561843, ...
            0.06250000000000, 0.04960628287401, 0.03937253280921, ...
            0.03125000000000, 0.02480314143700, 0.01968626640461, ...
            0.01562500000000, 0.01240157071850, 0.00984313320230, ...
            0.00781250000000, 0.00620078535925, 0.00492156660115, ...
            0.00390625000000, 0.00310039267963, 0.00246078330058, ...
            0.00195312500000, 0.00155019633981, 0.00123039165029, ...
            0.00097656250000, 0.00077509816991, 0.00061519582514, ...
            0.00048828125000, 0.00038754908495, 0.00030759791257, ...
            0.00024414062500, 0.00019377454248, 0.00015379895629, ...
            0.00012207031250, 0.00009688727124, 0.00007689947814, ...
            0.00006103515625, 0.00004844363562, 0.00003844973907, ...
            0.00003051757813, 0.00002422181781, 0.00001922486954, ...
            0.00001525878906, 0.00001211090890, 0.00000961243477, ...
            0.00000762939453, 0.00000605545445, 0.00000480621738, ...
            0.00000381469727, 0.00000302772723, 0.00000240310869, ...
            0.00000190734863, 0.00000151386361, 0.00000120155435];

% Rapport signal/bruit en fonction du nombre de pas de quantification
% La troisième colonne donne le nombre de bits nécessaire
% Pour n=3,5,7,9 on "groupe" par 3 => ceil(log2(n^3))/3

% pour les sous-bandes 0,1 et 2
Rsb_iso_1_3 = [0  0.00 0
               3  7.00 1.67
               7 16.00 3
              15 25.28 4
              31 31.59 5
              63 37.75 6
             127 43.84 7
             255 49.89 8
             511 55.93 9
            1023 61.96 10
            2047 67.98 11
            4095 74.01 12
            8191 80.03 13
           16383 86.05 14
           32767 92.01 15
           65535 98.01 16];

% pour les sous-bandes 3 à 10
```

```
Rsb_iso_4_11 = [0  0.00 0
                3  7.00 1.67
                5 11.00 2.33
                7 16.00 3
                9 20.84 3.33
               15 25.28 4
               31 31.59 5
               63 37.75 6
              127 43.84 7
              255 49.89 8
              511 55.93 9
             1023 61.96 10
             2047 67.98 11
             4095 74.01 12
             8191 80.03 13
            65535 98.01 16];

% pour les sous-bandes 11 à 22
Rsb_iso_12_23 = [0  0.00 0
                 3  7.00 1.67
                 5 11.00 2.33
                 7 16.00 3
                 9 20.84 3.33
                15 25.28 4
                31 31.59 5
             65535 98.01 16];

% pour les sous-bandes 23 à 26
Rsb_iso_24_27 = [0  0.00 0
                 3  7.00 1.67
                 5 11.00 2.33
             65535 98.01 16];

A_B = [3 0.750000000 -0.250000000
       7 0.875000000 -0.125000000
      15 0.937500000 -0.062500000
      31 0.968750000 -0.031250000
      63 0.984375000 -0.015625000
     127 0.992187500 -0.007812500
     255 0.996093750 -0.003906250
     511 0.998046875 -0.001953125
    1023 0.999023438 -0.000976563
    2047 0.999511719 -0.000488281
    4095 0.999755859 -0.000244141
    8191 0.999877930 -0.000122070
   16383 0.999938965 -0.000061035
   32767 0.999969482 -0.000030518];

function init_mpa1
```

```
% Définitions de constantes utilisées dans le modèle
% psychoacoustique no 1 de mpeg

global Fe LTq_i LTq_k Table_z Frontieres_i Frontieres_k Larg_f

% Détermination des indices "k" en fonction des indices "i"
i1 = 1:48;
i2 = (49:72)-48;
i3 = (73:108)-72;
i_to_k = [i1 (2*i2+48) (4*i3+96)];

% Seuil d'audition absolue en fonction des indices "i"
LTq_i = [25.87 14.85 10.72  8.50  7.10  6.11  5.37  4.79 ...
          4.32  3.92  3.57  3.25  2.95  2.67  2.39  2.11 ...
          1.83  1.53  1.23  0.90  0.56  0.21 -0.17 -0.56 ...
         -0.96 -1.38 -1.79 -2.21 -2.63 -3.03 -3.41 -3.77 ...
         -4.09 -4.37 -4.60 -4.78 -4.91 -4.97 -4.98 -4.92 ...
         -4.81 -4.65 -4.43 -4.17 -3.87 -3.54 -3.19 -2.82 ...
         -2.06 -1.32 -0.64 -0.04  0.47  0.89  1.23  1.51 ...
          1.74  1.93  2.11  2.28  2.46  2.63  2.82  3.03 ...
          3.25  3.49  3.74  4.02  4.32  4.64  4.98  5.35 ...
          6.15  7.07  8.10  9.25 10.54 11.97 13.56 15.31 ...
         17.23 19.34 21.64 24.15 26.88 29.84 33.05 36.52 ...
         40.25 44.27 48.59 53.22 58.18 63.49 68.00 68.00 ...
         68.00 68.00 68.00 68.00 68.00 68.00 68.00 68.00 ...
         68.00 68.00 68.00 68.00];

% Seuil d'audition absolue en fonction des indices "k"
 LTq_k = zeros(250, 1);
 for i = 1:length(LTq_i)
    LTq_k(i_to_k(i)) = LTq_i(i);
 end
 for k = 1:250
    if LTq_k(k) == 0
        LTq_k(k) = last_nonzero;
    else
        last_nonzero = LTq_k(k);
    end
 end

 % Fréquences Bark en fonction des indices "i"
 Table_z = [ .850 1.694 2.525 3.337 4.124 4.882 5.608 6.301 ...
           6.959  7.581  8.169  8.723  9.244  9.734 10.195 10.629 ...
          11.037 11.421 11.783 12.125 12.448 12.753 13.042 13.317 ...
          13.578 13.826 14.062 14.288 14.504 14.711 14.909 15.100 ...
          15.284 15.460 15.631 15.796 15.955 16.110 16.260 16.406 ...
          16.547 16.685 16.820 16.951 17.079 17.205 17.327 17.447 ...
          17.680 17.905 18.121 18.331 18.534 18.731 18.922 19.108 ...
          19.289 19.464 19.635 19.801 19.963 20.120 20.273 20.421 ...
          20.565 20.705 20.840 20.972 21.099 21.222 21.342 21.457 ...
```

```
          21.677 21.882 22.074 22.253 22.420 22.576 22.721 22.857 ...
          22.984 23.102 23.213 23.317 23.415 23.506 23.592 23.673 ...
          23.749 23.821 23.888 23.952 24.013 24.070 24.125 24.176 ...
          24.225 24.271 24.316 24.358 24.398 24.436 24.473 24.508 ...
          24.542 24.574 25 25];

Frontieres_i = [1 2 3 5 6 8 9 11 13 15 17 20 23 27 32 37 ...
         45 50 55 61 68 75 81 93 106];

Frontieres_k = zeros(1, length(Frontieres_i));
for i = 1:length(Frontieres_i)
   Frontieres_k(i) = i_to_k(Frontieres_i(i));
end

% Détermination de la "largeur" des bandes critiques
f_250_c = [0 Frontieres_k 296];
f_250_d = [0 Frontieres_k 256];

upper_bound = 1;
lower_bound = 1;
while upper_bound < 250
   lower_bound = lower_bound + 1;
   for k = 1:25
       if lower_bound >= f_250_c(k) & lower_bound < f_250_c(k+1)
           larg_bas = f_250_c(k+1) - f_250_c(k);
           no_ech_bas = f_250_c(k+1) - lower_bound;
       end
   end
   if no_ech_bas >= ceil(larg_bas/2)
       larg_fen = ceil(larg_bas/2);
   else
       for k = 1:25
           if upper_bound >= f_250_c(k) & upper_bound < f_250_c(k+1)
               larg_haut = f_250_c(k+1) - f_250_c(k);
               no_ech_haut = upper_bound - f_250_c(k);
           end
       end
       no_ech_tot = no_ech_haut + no_ech_bas;
       larg_fen = ceil((larg_bas*no_ech_bas/no_ech_tot+larg_haut*no_ech_haut/no_ech_tot)/2);
   end
   upper_bound = lower_bound + larg_fen;
   Larg_f(lower_bound) = larg_fen;
end

function [seuil1_db, seuil2_db, SMR_db_sb] = mpa1_mpeg(xn_mpa)

% Calcul du seuil de masquage correspondant au modèle psychoacoustique
% no 1 de mpeg (cf pages 122-128 de la norme ISO/CEI 11172-3:1993 (F))
% Entrees
%    xn_mpa(512,1) : signal non pondéré
```

```
%      plus quelques constantes définies dans init_mpa1_mpeg.m
% Sorties
%    seuil1_db(1,256) : seuil de masquage exprime en db
%    seuil2_db(1,256) : min du precedent dans chacune des 32 sous-bandes
%      veritable seuil utilise par le codeur mpeg pour réaliser son
%      allocation binaire
%    SMR_db_sb(1,32)  : 27 rapports signal/masque,
%      les autres ne seront pas exploites
%

global Fe LTq_i LTq_k Table_z Frontieres_i Frontieres_k Larg_f

% Etape 1 : Analyse FFT
% *********************
N = length(xn_mpa);
hann = sqrt(8/3)/2*[ones(N, 1) - cos(2*pi*(0:N-1)'/N)];
if sum(abs(xn_mpa)) > 0
   X1 = fft(xn_mpa.*hann);
   X1 = (abs(X1(1:N/2+1)).^2)/N;
   perio_xn_db = 10*log10(X1);
else
   perio_xn_db = zeros(N/2+1,1);
end
offset = max(perio_xn_db) - 96;
X = perio_xn_db - offset;

% Etape 2 : Determination du niveau de pression acoustique
% ********************************************************
% Non programmée

% Etape 3 : Prise en compte du seuil d'audition absolu
% ****************************************************
% Non programmée

% Etape 4 : Determination des composantes tonales et non-tonales
% **************************************************************

% Recherche des maxima locaux
max_local = zeros(250, 1);
for k = 3:250
   if X(k) > X(k-1) & X(k) >= X(k+1)
       max_local(k) = 1;
   end
end

tonal = zeros(250, 1);
for k = 3:62
   if max_local(k)
       tonal(k) = 1;
       for j = [-2 2]
```

```
            if X(k) - X(k+j) < 7
                tonal(k) = 0;
            end
        end
    end
end

for k = 63:126
    if max_local(k)
        tonal(k) = 1;
        for j = [-3 -2 2 3]
            if X(k) - X(k+j) < 7
                tonal(k) = 0;
            end
        end
    end
end

for k = 127:250
    if max_local(k)
        tonal(k) = 1;
        for j = [-6:-2 2:6]
            if X(k) - X(k+j) < 7
                tonal(k) = 0;
            end
        end
    end
end

% Recherche des raies tonales
X_tm = zeros(250,1);
for k = 1:250
    if tonal(k)
        temp = 10^(X(k-1)/10) + 10^(X(k)/10) + 10^(X(k+1)/10);
        X_tm(k) = 10*log10(temp);
        X(k-1) = -100;
        X(k)   = -100;
        X(k+1) = -100;
    else
        X_tm(k) = -100;
    end
end

X_nm = -100*ones(250, 1);
k = 1;
for k1 = Frontieres_k
    geom_mean = 1;
    pow = 0;
    raies_en_sb = 0;
    while k <= k1
```

```
        geom_mean = geom_mean*k;
        pow = pow + 10^(X(k)/10);
        k = k + 1;
        raies_en_sb = raies_en_sb + 1;
    end
    geom_mean = floor(geom_mean^(1/raies_en_sb));
    X_nm(geom_mean) = 10*log10(pow);
end

X_tm_avant = X_tm;
X_nm_avant = X_nm;

% Etape 5 : Decimation des composantes masquantes
% ***********************************************

for k = 1:250
    if X_tm(k) < LTq_k(k)
        X_tm(k) = -100;
    end
    if X_nm(k) < LTq_k(k)
        X_nm(k) = -100;
    end
end

% Elimination des raies tonales voisines: fenetre glissante
upper_bound = 1;
lower_bound = 1;
while upper_bound < 250
    [ans, max_ix] = max(X_tm(lower_bound:upper_bound));
    for k = lower_bound:upper_bound
        if k-lower_bound+1 ~= max_ix
            X_tm(k) = -100;
        end
    end
    lower_bound = lower_bound + 1;
    upper_bound = lower_bound + Larg_f(lower_bound);
end

% Etape 6 : Calcul des seuils de masquage individuels
% ***************************************************

% decimation : indices "k" vers indices "i"
Nbre_comp_i = length(Table_z);
X_tm_i = -100*ones(Nbre_comp_i, 1);
X_nm_i = -100*ones(Nbre_comp_i, 1);

for k = 1:250
    if X_tm(k) >= -10
        X_tm_i(ppv(k)) = X_tm(k);
    end
```

```
end

for k = 1:250
   if X_nm(k) >= -10
       X_nm_i(ppv(k)) = X_nm(k);
   end
end

% Etape 6->7 : Calcul des seuils de masquage globaux
% **************************************************

seuil_m = zeros(Nbre_comp_i, 1);
no_tm = 0;
no_nm = 0;
for i = 1:Nbre_comp_i
   if X_tm_i(i) > -100
       no_tm = no_tm + 1;
   end
   if X_nm_i(i) > -100
       no_nm = no_nm + 1;
   end
end

tab_tm = zeros(1,no_tm);
tab_nm = zeros(1,no_nm);
ix = 1;
for i = 1:Nbre_comp_i
   if X_tm_i(i) > -100
       tab_tm(ix) = i;
       ix = ix + 1;
   end
end

ix = 1;
for i = 1:Nbre_comp_i
   if X_nm_i(i) > -100
       tab_nm(ix) = i;
       ix = ix + 1;
   end
end

for i = 1:Nbre_comp_i
   sum_tm = 0;
   z_i = Table_z(i);
   for j = tab_tm
       z_j = Table_z(j);
       dz = z_i - z_j;
       if dz >= -3 & dz < 8
           LT_tm = X_tm_i(j) + (-1.525 - 0.275*z_j - 4.5) + vf(dz, j, X_tm_i);
           sum_tm = sum_tm + 10 ^ (LT_tm/10);
```

```
        end
    end
    sum_nm = 0;
    for j = tab_nm
        z_j = Table_z(j);
        dz = z_i - z_j;
        if dz >= -3 & dz < 8
            LT_nm = X_nm_i(j) + (-1.525 - 0.175*z_j - 0.5) + vf(dz, j, X_nm_i);
            sum_nm = sum_nm + 10 ^ (LT_nm/10);
        end
    end
    seuil_m(i) = 10 * log10(10^(LTq_i(i)/10) + sum_tm + sum_nm);
end

% Etape 8 : Determination
%           - du seuil de masquage
%           - du seuil de masquage minimum dans chaque sous-bande
%           - des rapports signal sur masque
% ************************************************************

seuil1_db = zeros(1,256);
seuil2_db = zeros(1,256);
for i = 1:6
    t1 = seuil_m(8*(i-1)+1:8*(i));
    seuil1_db(8*(i-1)+1:8*(i)) = t1;
    seuil2_db(8*(i-1)+1:8*(i)) = ones(1,8)*min(t1);
end
for i = 7:12
    i1 = i - 6;
    t1 = seuil_m(49+4*(i1-1):48+4*(i1));
    t2(1:2:7) = t1;
    t2(2:2:8) = t1;
    seuil1_db(8*(i-1)+1:8*(i)) = t2;
    seuil2_db(8*(i-1)+1:8*(i)) = ones(1,8)*min(t1);
end
for i = 13:30
    i1 = i - 12;
    t1 = seuil_m(73+2*(i1-1):72+2*(i1));
    t2(1:4:5) = t1;
    t2(2:4:6) = t1;
    t2(3:4:7) = t1;
    t2(4:4:8) = t1;
    seuil1_db(8*(i-1)+1:8*(i)) = t2;
    seuil2_db(8*(i-1)+1:8*(i)) = ones(1,8)*min(t1);
end
for i = 31:32
    seuil1_db(8*(i-1)+1:8*(i)) = ones(1,8)*min(t1);
    seuil2_db(8*(i-1)+1:8*(i)) = ones(1,8)*min(t1);
end
```

```
seuil1_db = seuil1_db + offset;
seuil2_db = seuil2_db + offset;
SMR_db_sb = zeros(1,32);
for i = 1:32
   SMR_db_sb(i) = max(perio_xn_db((i-1)*8+1:i*8)) - seuil2_db(i*8);
end

function i0 = ppv(k0)
if k0 <= 48
   i0 = k0;
elseif k0 <= 96
   i0 = floor((k0-48)/2) + 48;
else
   i0 = round((k0-96)/4) + 72;
end
if i0 > 108
   i0 = 108;
end

function le_vf = vf(dz, j, X)
if dz < -1
   le_vf = 17 * (dz + 1) - (0.4 * X(j) + 6);
elseif dz < 0
   le_vf = (0.4 * X(j) + 6) * dz;
elseif dz < 1
   le_vf = -17 * dz;
else
   le_vf = -(dz - 1) * (17 - 0.15 * X(j)) - 17;
end

function NOQ = mpeg_alloc(SMR, Debit_kbs)

% Procédure d'allocation de bits

global Fe
global Rsb_iso_1_3 Rsb_iso_4_11 Rsb_iso_12_23 Rsb_iso_24_27

SMR = [SMR(1:27) zeros(1,5)];
Bits_disp_init = Debit_kbs*1000*384/Fe - 88;
Bits_disp = Bits_disp_init;
NOQ = ones(1,27);
NOQ_max = [15*ones(1,11) 7*ones(1,12) 3*ones(1,4)];
MNR = -SMR(1:27);
MNR_init = MNR;
SNR = zeros(1,27);
Nbre_bits = zeros(1,27);
Scf = zeros(1,27);

iter = 0;
iter_max = 1000;
```

```
while Bits_disp > 0
   [temp kmin] = min(MNR);
   if NOQ(kmin) == NOQ_max(kmin)
       MNR(kmin) = 100;
   else
       Scf(kmin) = 1;
       NOQ(kmin) = NOQ(kmin) + 1;
       if kmin < 4
           SNR(kmin)       = Rsb_iso_1_3(NOQ(kmin),2);
           Nbre_bits(kmin) = Rsb_iso_1_3(NOQ(kmin),3);
       elseif kmin < 12
           SNR(kmin)       = Rsb_iso_4_11(NOQ(kmin),2);
           Nbre_bits(kmin) = Rsb_iso_4_11(NOQ(kmin),3);
       elseif kmin < 24
           SNR(kmin)       = Rsb_iso_12_23(NOQ(kmin),2);
           Nbre_bits(kmin) = Rsb_iso_12_23(NOQ(kmin),3);
       elseif kmin < 28
           SNR(kmin)       = Rsb_iso_24_27(NOQ(kmin),2);
           Nbre_bits(kmin) = Rsb_iso_24_27(NOQ(kmin),3);
       end
       MNR(kmin) = SNR(kmin) - SMR(kmin);
       Bits_disp_old = Bits_disp;
       Bits_disp = Bits_disp_init - 6*sum(Scf) - 12*sum(Nbre_bits);
   end
   iter = iter + 1;
   if iter > iter_max
       fprintf('Dans mpeg_alloc Bits_disp_init = %d Bits_disp = %d \n', Bits_disp_init, Bits
       return
   end
end

function ecrit_bit_stream(fid_bit_stream, NOQ, Mat_sb)

% Quantification de la matrice des signaux de sous-bande
% et construction du bit stream

global Fact_ech Rsb_iso_1_3 Rsb_iso_4_11 Rsb_iso_12_23 Rsb_iso_24_27

% On écrit d'abord l'allocation binaire
for k = 1:11
   fwrite(fid_bit_stream, NOQ(k)-1, 'ubit4');
end
for k = 12:23
   fwrite(fid_bit_stream, NOQ(k)-1, 'ubit3');
end
for k = 24:27
   fwrite(fid_bit_stream, NOQ(k)-1, 'ubit2');
end
for k = 1:27
   if k < 4
```

```
        Nbre_pas_quantif(k) = Rsb_iso_1_3(NOQ(k),1);
    elseif k < 12
        Nbre_pas_quantif(k) = Rsb_iso_4_11(NOQ(k),1);
    elseif k < 24
        Nbre_pas_quantif(k) = Rsb_iso_12_23(NOQ(k),1);
    elseif k < 28
        Nbre_pas_quantif(k) = Rsb_iso_24_27(NOQ(k),1);
    end
 end

% On écrit ensuite les facteurs d'échelle
 for k = 1:27
    if Nbre_pas_quantif(k) > 0
        vmax = max(abs(Mat_sb(:,k)));
        indice_Scf = 63;
        while vmax > Fact_ech(indice_Scf) & indice_Scf >= 2
            indice_Scf = indice_Scf - 1;
        end
        if vmax > Fact_ech(1)
            indice_Scf = 1;
        end
        fwrite(fid_bit_stream, indice_Scf, 'ubit6');
        Scf(k) = Fact_ech(indice_Scf);
    end
 end

 % On écrit enfin les échantillons de sous-bandes
 for k = 1:27
    if Nbre_pas_quantif(k) > 0
        alpha = 2*Scf(k)/Nbre_pas_quantif(k);
        dico = -Scf(k)+alpha/2:alpha:Scf(k)-alpha/2;
        indices_ech = zeros(12, 1);
        for n = 1:12
            [val indice] = min(abs(dico - Mat_sb(n,k)));
            indices_ech(n) = indice;
        end

        if Nbre_pas_quantif(k) <= 9
            nbre_bits_prec = ceil(log2(Nbre_pas_quantif(k)^3));
            precision = ['ubit' int2str(nbre_bits_prec)];
            carre = Nbre_pas_quantif(k)^2;
            for n = 1:4
                n1 = (n-1)*3 + 1;
                ind = indices_ech(n1)-1 + ...
                    (indices_ech(n1+1)-1)*Nbre_pas_quantif(k) + ...
                    (indices_ech(n1+2)-1)*carre;
                fwrite(fid_bit_stream, ind, precision);
            end
        else
            nbre_bits_prec = log2(Nbre_pas_quantif(k)+1);
```

```
            precision = ['ubit' int2str(nbre_bits_prec)];
            for n = 1:12
                fwrite(fid_bit_stream, indices_ech(n)-1, precision);
            end
        end
    end
end

function [Mat_sb_q, stop] = lit_bit_stream(fid_bit_stream)

% Lecture du bit stream et reconstruction de la matrice des signaux de sous-bande

global Fact_ech Rsb_iso_1_3 Rsb_iso_4_11 Rsb_iso_12_23 Rsb_iso_24_27

Mat_sb_q = zeros(12,32);
stop = 0;

% On lit d'abord les allocations binaires
for k = 1:11
    [val nbre] = fread(fid_bit_stream, 1, 'ubit4');
    if nbre == 0
        stop = 1;
        return
    end
    NOQ(k) = val + 1;
    if k < 4
        Nbre_pas_quantif(k) = Rsb_iso_1_3(NOQ(k),1);
    elseif k < 12
        Nbre_pas_quantif(k) = Rsb_iso_4_11(NOQ(k),1);
    end
end
for k = 12:23
    [val nbre] = fread(fid_bit_stream, 1, 'ubit3');
    if nbre == 0
        stop = 1;
        return
    end
    NOQ(k) = val + 1;
    Nbre_pas_quantif(k) = Rsb_iso_12_23(NOQ(k),1);
end
for k = 24:27
    [val nbre] = fread(fid_bit_stream, 1, 'ubit2');
    if nbre == 0
        stop = 1;
        return
    end
    NOQ(k) = val + 1;
    Nbre_pas_quantif(k) = Rsb_iso_24_27(NOQ(k),1);
end
```

```
% Ensuite les facteurs d'échelle
for k = 1:27
   if Nbre_pas_quantif(k) > 0
       indice_Scf = fread(fid_bit_stream, 1, 'ubit6');
       Scf(k) = Fact_ech(indice_Scf);
   end
end

% Enfin les échantillons de sous-bandes
for k = 1:27
   if Nbre_pas_quantif(k) > 0
       alpha = 2*Scf(k)/Nbre_pas_quantif(k);
       dico = -Scf(k)+alpha/2:alpha:Scf(k)-alpha/2;
       if Nbre_pas_quantif(k) <= 9
           nbre_bits_prec = ceil(log2(Nbre_pas_quantif(k)^3));
           precision = ['ubit' int2str(nbre_bits_prec)];
           carre = Nbre_pas_quantif(k)^2;
           for n = 1:4
               ind = fread(fid_bit_stream, 1, precision);
               n1 = (n-1)*3 + 1;
               ind3 = fix(ind/carre);
               ind2 = fix((ind - carre*ind3)/Nbre_pas_quantif(k));
               ind1 = ind - carre*ind3 - Nbre_pas_quantif(k)*ind2;
               Mat_sb_q(n1,k) = dico(ind1+1);
               Mat_sb_q(n1+1,k) = dico(ind2+1);
               Mat_sb_q(n1+2,k) = dico(ind3+1);
           end
       else
           nbre_bits_prec = log2(Nbre_pas_quantif(k)+1);
           precision = ['ubit' int2str(nbre_bits_prec)];
           for n = 1:12
               indice_ech = fread(fid_bit_stream, 1, precision);
               Mat_sb_q(n,k) = dico(indice_ech+1);
           end
       end
   else
       Scf(k) = 0;
       Mat_sb_q(:,k) = zeros(12,1);
   end
end
```

Bibliographie

[1] S. Lipshitz, R. Wannamaker, and J. Vanderkooy, "Quantization and dither : a theoretical survey," *J. Audio Eng. Soc.*, vol. 40, no. 5, pp. 355–375, May 1992.

[2] N. Jayant and P. Noll, *Digital coding of waveforms.* Prentice Hall, 1984.

[3] A. Gersho and R. Gray, *Vector quantization and signal compression.* Kluwer Academic Publishers, 1992.

[4] S. Kay, *Modern spectral estimation - Theory and application.* Prentice Hall, 1988.

[5] R. Gray, *Source coding theory.* Kluwer Academic Publishers, 1990.

[6] N. Sloane, "Les empilements de sphères," *Pour la Science*, pp. 44–55, Mars 1984.

[7] Y. Linde, A. Buzo, and R. Gray, "An algorithm for vector quantizer design," *IEEE Trans. on Communications*, vol. COM-28, January 1980.

[8] P. Zador, "Asymptotic quantization error of continuous signals and the quantization dimension," *IEEE Trans. on Information Theory, vol. IT-28*, pp. 139–149, March 1982.

[9] H. Malvar, *Signal processing with lapped transforms.* Artech House, 1992.

[10] P. Vaidyanathan, "Multirate digital filters, filter banks, polyphase networks and applications : A tutorial," *Proceedings of the IEEE*, January 1990.

[11] O. Rioul, *Ondelettes régulières : application à la compression d'images fixes.* PhD thesis, ENST, Mars 1993.

[12] Y. Shoham and A. Gersho, "Efficient bit allocation for an arbitrary set of quantizers," *IEEE Trans. on Acoust., Speech, and Signal Processing*, vol. 36, no. 9, pp. 1445–1453, September 1988.

[13] S. Haykin, *Adaptive filter theory.* Prentice Hall, 1991 (2eme édition).

[14] R. Gray, "On the asymptotic eigenvalue distribution of toeplitz matrices," *IEEE Trans. on Information Theory*, vol. IT-18, no.6, pp. 725–730, November 1972.

[15] C. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423 and 623–656, 1948.

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

[17] R. Gallager, *Information theory and reliable communication.* Wiley, 1968.

[18] T. Berger, *Rate-distortion theory : A mathematical basis for data compression.* Prentice-Hall, 1971.

[19] R. Blahut, *Principles and practice of information theory.* Addison-Wesley, 1987.

[20] T. Cover and J. Thomas, *Elements of information theory.* Wiley series in Telecommunications, 1991.

[21] P. Howard and J. Vitter, "Arithmetic coding for data compression," *Processing of the IEEE*, vol. 82, no. 6, pp. 857–865, 1994.

[22] CCITT, *Normalisation des télécopieurs du groupe 3 pour la transmission de documents, Fascicule VII.3, Recommandation T.4*, 1980.

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

[24] European Telecommunication Standard, *Digital cellular telecommunications system ; Enhanced Full Rate (EFR) speech transcoding (GSM 06.60)*, Final Draft 1996.

[25] B. Bessette, R. Salami, R. Lefebvre, M. Jelinek, J. Rotola-Pukkila, J. Vainio, H. Mikkola, and K. Jarvinen, "The adaptive multirate wideband speech codec (AMR-WB)," *IEEE Trans. on Speech and Audio Processing*, vol. 10, no. 8, pp. 620–636, November 2002.

[26] K. Konstantinides, "An introduction to Super Audio CD and DVD-Audio," *IEEE Signal Processing Magazine*, July 2003.

[27] Norme internationale ISO/CEI 11172, *Codage de l'image animée et du son associé pour les supports de stockage numérique jusqu'à environ 1,5 Mbit/s*, 1993.

[28] International Organization for Standardization, *ISO/IEC 13818-7 (MPEG-2 Advanced Audio Coding, AAC)*, 1997.

[29] International Organization for Standardization, *ISO/IEC 14496-3 (Information technology - Very low bitrate audio-visual coding)*, 1998.

[30] International Organization for Standardization, *ISO/IEC 14496-3 :2005 (Information technology - Coding of audio-visual objects)*, 2005.

[31] P. Collen, *Techniques d'enrichissement de spectre des signaux audionumériques*. PhD thesis, ENST, Novembre 2002.

[32] *http ://en.wikipedia.org/wiki/AacPlus*.

[33] Union Internationale des Télécommunications, *Recommandation BS.1387 : Méthode de mesure objective de la qualité du son perçu*, 2001.

[34] B. Atal and J. Remde, "A new model of lpc excitation for producing natural-sounding speech at low bit rates," *Proc. Int. Conf. Acoust., Speech, Signal Processing*, pp. 614–617, 1982.

[35] M. Schroeder and B. Atal, "Code-excited linear prediction (CELP) : high-quality speech at very low bit rates," *Proc. Int. Conf. Acoust., Speech, Signal Processing*, pp. 937–940, 1985.

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

[37] B. Atal and M. Schroeder, "Predictive coding of speech signals and subjective error criteria," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-27, June 1979.

[38] G. Golub and C. Van Loan, *Matrix computations*. Johns Hopkins University Press, 1983 (Second Edition 1989).

[39] Union Internationale des Télécommunications, *Recommandation G.729 : Codage de la parole à 8 kbit/s par prédiction linéaire avec excitation par séquences codées à structure algébrique conjuguée*, 1996.

[40] R. Salami, C. Laflamme, J. Adoul, A. Kataoka, S. Hayashi, C. Lamblin, D. Massaloux, S. Proust, P. Kroon, and Y. Shoham, "Design and description of CS-ACELP : a toll quality 8 kb/s speech coder," *IEEE Trans. on Speech and Audio Processing*, vol. 6, no. 2, pp. 116–130, March 1998.

[41] T. Painter and A. Spanias, "Perceptual coding of digital audio," *Proceedings of the IEEE*, vol. 88, April 2000.

[42] O. Derrien, S. Larbi, M. Perreau-Guimaraes, and N. Moreau, "Le codeur MPEG-2 AAC expliqué aux traiteurs de signaux," *Annales des Télécommunications*, Septembre-Octobre 2000.

[43] Advanced television systems comittee, *Digital audio compression standard (AC-3)*, 1995.

[44] E. Zwicker and E. Feldtkeller, *Psychoacoustique, l'oreille récepteur d'information.* Masson, Collection technique et scientifique des télécommunications, 1981.

[45] B. Moore, *An introduction to the psychology of hearing.* Academic Press, Second edition, 1982.

[46] A. Goyé, *La Perception Auditive.* Polycopié ENST, 2002.

[47] F. Baumgarte and C. Faller, "Binaural cue coding - Part I : Psychoacoustic fundamentals and design principles," *IEEE Trans. on Speech and Audio Processing*, vol. 11, no. 6, pp. 509–519, November 2003.

[48] C. Faller and F. Baumgarte, "Binaural cue coding - Part II : Schemes and applications," *IEEE Trans. on Speech and Audio Processing*, vol. 11, no. 6, pp. 520–531, November 2003.

[49] C. Faller, *Parametric coding of spatial audio.* PhD thesis, Ecole Polytechnique Fédérale de Lausanne, 2004.

[50] J. Breebaart, S. van de Par, A. Kohlrausch, and E. Schuijers, "Parametric coding of stereo audio," *EURASIP Journal on Applied Signal Processing*, pp. 1305–1322, 2005.

[51] M. Briand, *Etudes d'algorithmes d'extraction des informations de spatialisation sonore : application aux formats multicanaux.* PhD thesis, Institut National Polytechnique de Grenoble, Mars 2007.

[52] J. Lapierre and R. Lefebvre, "On improving parametric stereo audio coding," *120th Convention AES*, Paris, May 2006.

[53] J. Herre, "From joint stereo to spatial audio coding - recent progress and standardization," *Proc. of the 7th Int. Conference on Digital Audio Effects*, Naples, October 2004.

[54] H. Purnhagen, "Low complexity parametric stereo coding in MPEG-4," *Proc. of the 7th Int. Conference on Digital Audio Effects*, Naples, October 2004.

[55] O. Calvet, *Acoustique appliquée aux techniques du son.* Editions Casteilla, 2002.

[56] M. Hans and R. Schafer, "Lossless compression of digital audio," *IEEE Signal Processing Magazine*, July 2001.

[57] R. Geiger, R. Yu, J. Herre, S. Rahardja, S. Kim, X. Lin, and M. Schmidt, "ISO/IEC MPEG-4 High-Definition Scalable Advanced Audio Coding," *120th Convention AES, Paris*, May 2006.

[58] R. Yu, S. Rahardja, L. Xiao, and C. Ko, "A fine granular scalable to lossless audio coder," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 14, July 2006.

Contexte académique } avec modifications