CHAPITRE 4

Quantification et Codage entropique

1. Généralités; allocation de bits

La dernière étape du codage d'un signal, postérieure à la quantification, est le codage proprement dit. On désigne par le vocable codage un ensemble de techniques visant à associer des suites binaires (c'est à dire des suites de zéros et de uns) à un dictionnaire de mots. Dans le cas qui nous préoccupe, ces mots sont les valeurs (discrètes) de coefficients d'un signal dans une base donnée, après quantification. Cependant, les techniques de codage s'appliquent également à d'autres situations, telles que le codage de fichiers informatiques par exemple.

On a vu dans le cas des codeurs PCM et DPCM des exemples utilisant un code de longueur constante: tous les coefficients étaient codés sur un nombre R constant de bits, le code étant fixé une fois pour toutes. Cependant, il s'avère souvent plus efficace d'utiliser des codes de longueur variable, c'est à dire des codes dans lesquels on n'affecte pas nécessairement un code de la même longueur à différents symboles (dans notre cas, des coefficients quantifiés). On pourra ainsi affecter un code très court à des symboles très fréquent, et un code plus long à des symboles moins fréquents, afin d'optimiser le débit total. On rappelle que dans le cas d'un code de longueur constante, chaque symbole est codé sur R bits, ce qui correspond à $M = 2^R$ symboles.

Les codes de longueur variable posent des problèmes différents, qu'on va étudier ci dessous.

2. Codes de longueur variable

2.1. Introduction. Pour fixer les idées, on considère une suite de variables aléatoires X_n, prenant leurs valeurs dans un alphabet fini $A = \{a_0, a_1, \dots a_{M-1}\}$, avec des probabilités $\Pr X_n = a = p(a)$. Un codeur associera à chaque symbole $a \in A$ un mot binaire $\alpha(a)$, de longueur $\ell(a)$ (mesurée en bits). Le meilleur codeur sera celui qui minimise la longueur moyenne

$$\overline{\ell} = \sum_{a \in A} p(a)\ell(a) \ . \tag{4.1}$$

On rappelle que dans le cas d'un code de longueur constante, chaque symbole a_k est codé sur $\ell(a_k) = R$ bits, ce qui correspond à $M = 2^R$ symboles. Par conséquent, on a

$$\sum_{a \in A} 2^{-\ell(a)} = 1 \ . \tag{4.2}$$

On verra par la suite la signification de cette identité simple.

DÉFINITION 4.1. *Un code scalaire sans perte de longueur variable consiste en*

(1) *Un codeur: une application α qui associe à tout symbole d'entrée $a \in A$ une suite binaire $\alpha(a)$, de longueur $\ell(a)$.*
(2) *Un décodeur: une application β qui associe à toute suite binaire u un symbole $a = \beta(u) \in A$, tel que pour tout $a \in A$, $\beta(\alpha(a)) = a$.*

Le codeur est ensuite étendu aux suites de symboles (les mots) par concaténation: la suite binaire associée au mot $x_1x_2 \dots x_n$ est la concaténation

$$\alpha(x_1x_2 \dots x_n) = \alpha(x_1)\alpha(x_2) \dots \alpha(x_n) \ .$$

La concaténation pose des problèmes si le code est un code de longueur variable. Dans le cas général, on ne sait pas où commencent et où s'arrètent les mots. On introduit donc une sous-classe de codes, appelés codes uniquement décodables, définis comme suit.

DÉFINITION 4.2. *Le code est dit uniquement décodable si le codeur α est injectif: c'est à dire si toute suite binaire $\alpha(x_1x_2 \dots x_n)$ n'est l'image que d'un et un seul mot, à savoir $x_1x_2 \dots x_n$.*

EXEMPLE 4.1. On considère les deux codeurs α_1 et α_2 et α_3 définis par les tableaux donnés dans la TABLE 1. On vérifie immédiatement que le code donné dans le tableau de gauche n'est pas uniquement décodable, alors que les deux autres le sont. Par contre, le code du tableau du milieu n'est pas *instantané*, au sens où il faut attendre le début du mot suivant pour savoir si un mot est terminé: 11 peut être suivi de 1, auquel cas il s'agit d'un a_4, ou d'un 0, auquel cas il s'agit d'un a_3. Le code présenté dans le tableau de droite est quant à lui un code uniquement décodable instantané.

input	code
a_0	0
a_1	10
a_2	101
a_3	0101

input	code
a_0	0
a_1	01
a_2	011
a_3	111

input	code
a_0	0
a_1	10
a_2	110
a_3	111

TAB. 1. *Trois exemples de codeurs: celui de gauche n'est pas uniquement décodable, les deux autres le sont. Le codeur du centre n'est pas instantané.*

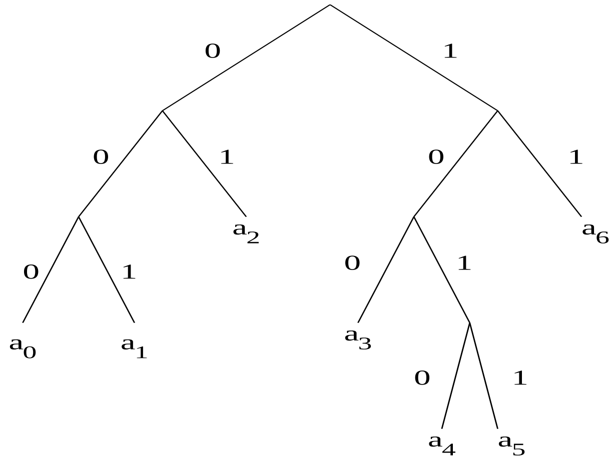

FIG. 1. *Exemple d'arbre binaire associé à un alphabet.*

input	a_0	a_1	a_2	a_3	a_4	a_5	a_6
code	000	001	01	100	1010	1011	11

TAB. 2. *Code associé à l'arbre binaire*

2.2. Codes à préfixe, et codes associés à un arbre binaire. Il existe une façon simple d'assurer qu'un code est uniquement décodable et instantané. Il suffit d'imposer une condition, appelée *condition de préfixe.*

DÉFINITION 4.3. *Un codeur α satisfait la condition de préfixe si aucun mot binaire $\alpha(a)$,$a \in A$ n'est préfixe d'un autre mot $\alpha(b)$,$b \in A$. Un code satisfaisant la condition de préfixe est appelé* code à préfixe.

Il existe une construction générique de codes à préfixes. On peut associer un tel code à tout arbre binaire. Plus précisément, étant donné un alphabet $A = \{a_0, \dots a_{M-1}\}$ à M symboles, on considère un arbre binaire à M feuilles. On associe à chaque symbole $a \in A$ une des feuilles, et on numérote les branches de l'arbre par des bits: par exemple, les branches partant vers la gauche sont notées "0", et les branches partant vers la droite sont notées "1". Le code associé à chaque symbole est alors la concaténation des étiquettes des branches consécutives menant de la racine de l'arbre à la feuille considérée.

3. Codes entropiques

3.1. L'inégalité de Kraft. On s'intéresse donc maintenant à des codes de longueur variable. On a vu que dans le cas de codes de longueur constante, si l'alphabet a $M = 2^R$ symboles, chaque mot $\alpha(a)$ est de longueur $\ell(a) = R$, de sorte que

$$\sum_{a \in A} 2^{-\ell(a)} = \sum_{a \in A} 2^{-R} = 1 \ .$$

On va voir que cette relation est une borne pour les codes de longueur variable uniquement décodables.

THÉORÈME 4.1. *Soit* $a = \{a_0, \dots a_{M-1}\}$ *un alphabet à* M *symboles. Le codeur* α*, qui associe à chaque* $a \in A$ *le mot* $\alpha(a)$ *de longueur* $\ell(a)$*, est uniquement décodable seulement si l'inégalité suivante, appelée* inégalité de Kraft*, est vérifiée:*

$$\sum_{a \in A} 2^{-\ell(a)} \leq 1 \ . \tag{4.3}$$

En fait, l'inégalité de Kraft montre que pour une distribution de probabilités de symboles donnée, les longueurs de mots binaires associés à ces symboles doivent, *dans une certaine moyenne*, être supérieures à une valeurs seuil. *Preuve:* Soit K un entier positif fixé. On considère une suite de K symboles $b = \{b_0, \dots, b_{K-1}\}$; la longueur totale de $\alpha(b_0 \dots b_{K-1})$ vaut

$$\ell(b) = \ell(b_0) + \cdots + \ell(b_{K-1}) \ .$$

On notera $N(L)$ le nombre de suites de K symboles ayant une longueur totale égale à L. On pose $\ell_{max} = \max_{a \in A} \ell(a)$. On a alors

$$\ell(b) \leq K\ell_{max} := L_{max} \ .$$

Le code étant uniquement décodable, les $N(L)$ suites ont un code différent. Comme il y a au plus 2^L codes de longueur L différents, on a

$$N(L) \leq 2^L \ .$$

Calculons alors

$$\begin{aligned}
\left[\sum_{m=0}^{M-1} 2^{-\ell(a_m)}\right]^K &= \left[\sum_{a \in A} 2^{-\ell(a)}\right]^K \\
&= \sum_{b_0 \in A} \sum_{b_1 \in A} \cdots \sum_{b_{K-1} \in A} 2^{-\ell(b_0) - \cdots - \ell(b_{K-1})} \\
&= \sum_{L=1}^{L_{max}} N(L) 2^{-L} \\
&\leq L_{max} = K\ell_{max} \ .
\end{aligned}$$

Par conséquent, on a

$$\sum_{m=0}^{M-1} 2^{-\ell(a_m)} \leq (K\ell_{max})^{1/K} \ .$$

Ceci étant vrai pour tout K, on obtient bien l'inégalité de Kraft par passage à la limite $K \to \infty$. Ceci conclut la preuve du théorème. ♠

Ce premier théorème fournit donc une condition nécessaire pour l'unique décodabilité d'un code. Cette condition est en fait également suffisante dans un certain sens, comme l'exprime le théorème suivant.

THÉORÈME 4.2. *Soit* $a = \{a_0, \dots a_{M-1}\}$ *un alphabet à* M *symboles, et soit* $\{\ell_0, \dots, \ell_{M-1}\}$ *une suite d'entiers positifs satisfaisant l'inégalité de Kraft (4.3). Alors il existe un codeur* α *uniquement décodable, tel que pour tout* $k = 0, \dots M-1$, $\ell(a_k) = \ell_k$.

3.2. L'entropie de Shannon. Considérons maintenant la longueur moyenne $\overline{\ell}(\alpha)$ des mots codés par un codeur α,

$$\overline{\ell}(\alpha) = \sum_{a \in A} p(a)\ell(a) \ . \tag{4.4}$$

et supposons que l'inégalité de Kraft soit vérifiée. On a alors

$$\begin{aligned}
\overline{\ell}(\alpha) &= -\sum_{a\in A} p(a) \log_2 \left(2^{-\ell(a)}\right) \\
&\geq -\sum_{a\in A} p(a) \log_2 \left(\frac{2^{-\ell(a)}}{\sum_{b\in A} 2^{-\ell(b)}}\right) \\
&\geq -\sum_{a\in A} p(a) \log_2 (q(a)) \ ,
\end{aligned}$$

où $q = \{q_0, \dots q_{M-1}\}$ est une autre distribution de probabilités, définie par

$$q_k = q(a_k) = \frac{2^{-\ell(a_k)}}{\sum_{b\in A} 2^{-\ell(b)}} \ . \tag{4.5}$$

Notons que cette inégalité devient une égalité si l'inégalité de Kraft est une égalité, c'est à dire lorsque pour tout $a \in A$, $q(a) = 2^{-\ell(a)}$.

On peut introduire l'entropie de la distribution de probabilités p:

$$H(p) = -\sum_{a\in A} p(a) \log_2(p(a)) \ . \tag{4.6}$$

L'entropie associée à une distribution de probabilités représente la quantité d'information reçue lorsqu'un évènement $a \in \mathscr{A}$ est observé. L'entropie possède nombre de propriétés simples. En particulier, on a

Lemme 4.1.

$$0 \leq H(p) \leq \log_2 \left(Card(A)\right) \ .$$

Preuve: L'entropie est positive par construction. Pour montrer la borne supérieure, il suffit de trouver la distribution de probabilités sur A qui maximise H (avec évidemment la contrainte $\sum_{a\in A} p(a) = 1$). En introduisant un multiplicateur de Lagrange pour imposer la contrainte, on se ramène à maximiser

$$-\sum_{a\in A} p(a) \log_2(p(a)) + \lambda \left(\sum_{a\in A} p(a) - 1\right)$$

par rapport aux nombres $p(a)$ et au multiplicateur λ. On voit facilement que la solution est donnée par $p(a) = 1/\mathrm{Card}(A)$, d'où le résultat. ♠

Ainsi, les faibles valeurs de l'entropie correspondent à des situations dans lesquelles un petit nombre de symboles sont très probables (donc pour lesquels $\log p(a)$ est faible) et un grand nombre de symboles sont très probables (donc pour lesquels $p(a)$ est faible).

On a aussi le résultat important suivant

Lemme 4.2. *Etant données deux distributions de probabilités p et q indexées par l'alphabet A, on a*

$$-\sum_{a\in A} p(a) \log_2(q(a)) \geq H(p) \ . \tag{4.7}$$

avec égalité si et seulement si les deux distributions sont identiques.

Preuve: On va utiliser le fait que $\log(x) \leq x - 1$, avec égalité si et seulement si $x = 1$. Calculons

$$\begin{aligned}\sum_{a\in A} p(a)\log_2\left(\frac{q(a)}{p(a)}\right) &= \frac{1}{\log(2)}\sum_{a\in A} p(a)\log\left(\frac{q(a)}{p(a)}\right)\\ &\leq \frac{1}{\log(2)}\sum_{a\in A} p(a)\left(\frac{q(a)}{p(a)} - 1\right)\\ &= 0\ ,\end{aligned}$$

avec égalité si et seulement si $p(a) = q(a)$ pour tout $a \in A$. Ceci prouve le lemme. ♠

En combinant ce résultat avec le précédent, on a donc la première partie du théorème suivant:

Théorème 4.3. *Soit $A = \{a_0, \dots, a_{M-1}\}$ un alphabet, et soit p la distribution de probabilités de ses symboles.*

(1) *Soit α un codeur dont les longueurs de mots satisfont à l'inégalité de Kraft. Alors on a*

$$\overline{\ell}(\alpha) \geq H(p)\ , \tag{4.8}$$

avec égalité si et seulement si $p(a) = 2^{-\ell(a)}$ pour tout $a \in A$.

(2) *Il existe un code uniquement décodable tel que*

$$\overline{\ell}(\alpha) < H(p) + 1\ . \tag{4.9}$$

Preuve: La première partie est une conséquence directe du lemme:

$$\overline{\ell}(\alpha) \geq -\sum_{a\in A} p(a)\log_2 q(a) \geq H(p)\ ,$$

et on a égalité (i.e. les deux inégalités intervenant dans celle-ci sont des égalités) si et seulement si $p(a) = q(a) = 2^{-\ell(a)}$ pour tout a. Pour la seconde partie, étant donnée la distribution p, soient les nombres ℓ_k définis par

$$2^{-\ell_k} \leq p(a_k) < 2^{1-\ell_k}\ .$$

Il est évident que $\sum_k 2^{-\ell_k} \leq \sum_k p(a_k) = 1$, donc l'inégalité de Kraft est satisfaite. Il existe donc un code uniquement décodable α associé aux longueurs de mots ℓ_k. De plus, on a

$$\ell_k < 1 - \log_2(p(a_k))\ ,$$

et donc

$$\overline{\ell}(\alpha) = \sum_k p(a_k)\ell_k < \sum_k p(a_k)(1 - \log_2(p(a_k))) = 1 + H(p)\ .$$

Ceci conclut la démonstration. ♠

On donne maintenant sans démonstration quelques résultats importants. Une démonstration peut être trouvée dans [**7, 9**].

Théorème 4.4. *Soit (α,β) un code de longueur variable, dont on note $\ell_k = \ell(a_k)$ les longueurs de mots. Il existe un code à préfixe qui a les mêmes longueurs de mots, et en particulier la même longueur moyenne $\overline{\ell}(\alpha)$.*

Définition 4.4. *Etant donné un alphabet A et une distribution de probabilités pour les symboles de A, on dit qu'un code α est optimal si il minimise la longueur moyenne des mots $\overline{\ell}(\alpha)$.*

Théorème 4.5. *Soit α un code à préfixe optimal.*

(1) *Si les symboles a et b sont tels que $p(a) > p(b)$, alors $\ell(a) \leq \ell(b)$.*
(2) *Les deux symboles les moins probables sont associés à des mots de même longueur, qui ne diffèrent que par leur bit le moins significatif.*

3.3. Le code de Huffman. Le code de Huffman est un algorithme récursif qui permet d'atteindre les performances optimales. Il est basé sur la règle d'agrégation suivante.

Partant d'un alphabet $A = \{a_0,a_1, \dots a_{M-1}\}$ dont les symboles ont probabilités $(p(a_0),p(a_1), \dots p(a_{M-1}))$, on suppose (sans perte de généralité)que les symboles sont ordonnés par probabilités décroissantes: $p(a_0) \geq p(a_1) \geq \dots$ (si nécessaire, on peut les ré-ordonner). On construit un nouvel alphabet de longueur $M-1$ en agrégeant les symboles a_{M-1} et a_{M-2} en un nouveau symbole $\tilde{a}$ auquel on affecte la probabilité

$$p(\tilde{a}) = p(a_{M-1}) + p(a_{M-2}) \ .$$

Théorème 4.6. *Un arbre de préfixe optimal pour l'alphabet à M symboles ordonnés $A = \{a_0,a_1, \dots a_{M-1}\}$ s'obtient à partir d'un arbre de préfixe optimal pour $\{a_0,a_1, \dots a_{M-3,\tilde{a}}\}$, en adjoignant à ce dernier deux branches liant $\tilde{a}$ aux symboles a_{M-2} et a_{M-1}.*

L'algorithme de Huffman exploite ce résultat de la façon suivante: a_{M-1} et a_{M-2} sont les feuilles extrêmes de l'arbre. Les deux branches correspondantes sont caractérisées par un bit.

On est donc en présence d'un nouvel alphabet de $M-1$ symboles

$$\{a_0,a_1, \dots a_{M-3},\tilde{a}\} \ .$$

Pour poursuivre l'algorithme, il suffit alors de réordonner ce nouvel alphabet par ordre de probabilités décroissantes, et d'itérer la procédure: prendre les deux (nouveaux) symboles de probabilités minimales, et leur associer deux nouvelles branches.

On en déduit

Corollaire 4.1. *Le code de Huffman est le code à préfixe optimal, et satisfait en particulier la borne (4.9).*

Exemple 4.2. Prenons l'exemple suivant d'un alphabet de 8 symboles, de probabilités données dans la table 3.

L'arbre de préfixes correspondant se trouve en Figure 2. On peut à partir de cet exemple estimer la longueur moyenne des mots:

$$\overline{\ell}(\alpha) = 0.25{\times}2{+}0.2{\times}2{+}0.2{\times}2{+}0.18{\times}3{+}0.09{\times}4{+}0.05{\times}5{+}0.02{\times}6{+}0.01{\times}6 = 2.63\,,$$

ce qui représente un gain par rapport à un codeur uniforme qui aurait nécessité 3 bits par symbole.

Remarque 4.1. On a vu au chapitre précédent l'exemple du codage de plages, et de son application au codage des facsimilés. Un fax consiste essentiellement en des plages de pixels noirs et des plages de pixels blancs. La norme pour le codage des fax considère des plages de longueur inférieure à 64 pixels seulement, de sorte qu'il est possible d'avoir à coder plusieurs plages blanches (ou noires) consécutives. Les distributions de probabilités des longueurs de plages blanches et noires ont été estimées expérimentalement, et ont conduit à un code de Huffman "standard" pour les longueurs de plages blanches et noires. Pour plus de détails, on pourra se référer à [**7**].

symbole	probabilité	code
a	0.01	110000
b	0.02	110001
c	0.05	11001
d	0.09	1101
e	0.18	111
f	0.2	00
g	0.2	01
h	0.25	10

Tab. 3. *Exemple de code de Huffman.*

Remarque 4.2. Il arrive souvent que le code optimal ne soit pas unique: pour une distribution de probabilités donnée, l'algorithme de Huffman peut laisser une certaine liberté dans les mots binaires à associer aux symboles (typiquement, lorsque deux probabilités sont égales). Il est alors facile de voir que bien que les codes soient différents, la longueur moyenne des mots $\overline{\ell}$ est la même, ce qui est la seule chose réellement importante.

Remarque 4.3. Il existe des alternatives au codage de Huffman. On peut en particulier mentionner le *codage arithmétique*, qui permet d'associer, dans un certain sens, des "nombre de bits non entiers" aux symboles de A. On montre que ceci permet d'améliorer légèrement les performances d'un codeur.

3.4. Codage par blocs. On a vu que le codage entropique (par exemple le code de Huffman) permet d'atteindre une longueur moyenne de mots presque optimale (essentiellement, comprise entre l'entropie et l'entropie +1). Cependant, lorsque l'entropie est faible, on peut ne plus s'en satisfaire. Le codage par blocs permet d'affiner le résultat.

Supposons que l'on ait à coder une suite de variables aléatoires i.i.d. (typiquement, des coefficients issus d'un codeur par transformation, quantifiés). L'idée est alors de ne plus coder les variablea aléatoires individuellement, mais plutôt de les regrouper par blocs de N variables aléatoires (formant donc des vecteurs aléatoires, dont les composantes sont i.i.d.). On peut facilement montrer le résultat suivant:

Lemme 4.3. *Soient $X_1,\dots X_N$ des variables aléatoires i.i.d., dont on note p la distribution de probabilités. Soit P la distribution de probabilités jointe de $X_1,\dots X_N$ (la loi produit). Alors on a*

$$H(P) = NH(p)\ . \tag{4.10}$$

On peut alors adopter la stratégie simple suivante. Etant donné un signal donc on connait des coefficients quantifiés $X_1,X_2,\dots$, on commence par les regrouper en

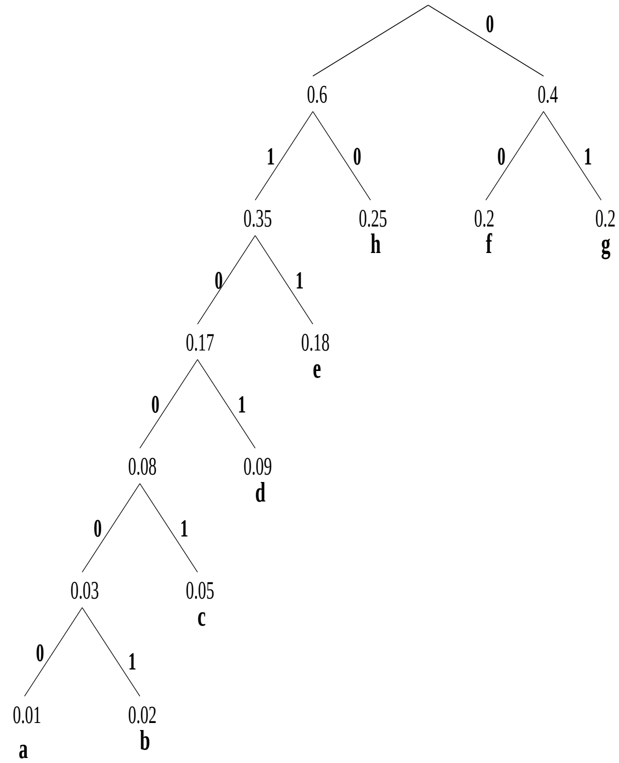

FIG. 2. *Arbre de préfixes pour les symboles de la Table 3*

blocs de N coefficients. A partir de la loi de ces blocs, on peut facilement construire un code entropique (le code de Huffman par exemple), qui fournit une longueur de mots moyenne telle que

$$\overline{\ell}_{\text{blocs}} < H(P) + 1 \ .$$

De là, si les blocs sont effectivement constitués de coefficients indépendants identiquement distribués, on en déduit que la longueur moyenne des mots associés aux coefficients (et non plus aux blocs) est telle que

$$\overline{\ell}_{\text{coeffs}} < H(p) + \frac{1}{N} \ .$$

Ainsi, en considérant des blocs de coefficients plutôt que des coefficients individuels, on peut s'approcher de la borne optimale. Par contre, il est clair que les algorithmes de codage et de décodage correspondants sont plus complexes.

4. Quantification non-uniforme et codage entropique

Dans les chapitres précédents, on n'avait abordé la quantification que dans une perspective de code de longueur constante. Il est maintenant utile de revenir sur ce problème de quantification à la lumière de ce que nous avons vu. On se limitera aux codes "haute résolution", ou "haut débit".

Supposons donc (pour simplifier) qu'on ait à quantifier une variable aléatoire de densité ρ_X à support borné (pour éviter d'avoir à considérer le bruit de saturation).

On considère donc comme précédemment les valeurs de quantification $y_0 < y_1 < \cdots < y_{M-1}$ et les bornes $x_0, \ldots x_M$, telles que

$$Q(x) = y_k \quad \text{si } x \in [x_k, x_{k+1}[\ .$$

Après quantification, les "intervalles de quantification" $[x_k, x_{k+1}[$ ont probabilité

$$p_k = \int_{x_k}^{x_{k+1}} \rho_X(x)\, dx \ .$$

La distorsion s'écrit alors

$$D = \sum_{k=0}^{M-1} \int_{x_k}^{x_{k+1}} (x - y_k)^2 \rho_X(x)\, dx \ .$$

Si on suppose comme précédemment que ρ_X varie peu dans chacun des intervalles $[x_k, x_{k+1}[$ (hypothèse de haute résolution), et si on pose

$$\Delta_k = x_{k+1} - x_k \ ,$$

et si on choisit $y_k \in [x_k, x_{k+1}[$ (par exemple $y_k = \frac{x_k + x_{k+1}}{2}$), on peut approximer, au premier ordre

$$p_k \approx \rho_X(y_k)\Delta_k \ ,$$

et

$$D \approx \sum_{k=0}^{M-1} \rho_X(y_k) \frac{\Delta_k^3}{12} \ .$$

Si on s'intéresse maintenant à l'entropie

$$\begin{aligned} H = -\sum_k p_k \log p_k &\approx -\sum_k \rho_X(y_k)\Delta_k \log \rho_X(y_k)\Delta_k \\ &\approx -\int \rho_X(x) \log \rho_X(x)\, dx - \frac{1}{2}\sum_k p_k \log \Delta_k^2 \ . \end{aligned}$$

Le premier terme est une caractéristique de la distribution de probabilités initiale de la variable aléatoire X, est appelé "entropie différentielle" de la source:

$$h(X) = -\int \rho_X(x) \log \rho_X(x)\, dx \ , \tag{4.11}$$

et caractérise la "concentration" de la variable aléatoire X.

Pour ce qui est du second terme, on peut utiliser la concavité du logarithme, qui donne l'inégalité de Jensen

$$\sum_k p_k \log \Delta_k^2 \leq \log \left(\sum_k p_k \Delta_k^2 \right) \ ,$$

et approximer

$$\frac{1}{2}\sum_k p_k \log \Delta_k^2 \leq \frac{1}{2} \log (12\, D) \ .$$

Mettant ensembles ces estimations, on aboutit à l'inégalité

$$H \geq h(X) - \frac{1}{2} \log (12\, D) \ ,$$

ou en d'autres termes

$$D \geq \frac{1}{12}\, 2^{-2[H - h(X)]} \ . \tag{4.12}$$

Ceci nous fournit une estimation de la distorsion minimale atteignable par le codeur considéré. On voit donc que pour améliorer les performances (optimales) d'un codeur (en termes de distorsion), on a intérêt à maximiser la différence entre l'entropie H et l'entropie différentielle de la source. En particulier, le résultat sera d'autant plus favorable que l'entropie différentielle $h(X)$ (qui caractérise la variabilité intrinsèque de la source) sera faible.

Cette expression est à comparer à l'expression analogue que nous avions obtenue dans le cas de la quantification uniforme:

$$D \approx C 2^{-2R} ,$$

avec laquelle elle coïncide dans le cas d'un code de longueur constante (toujours sous des hypothèses "haute résolution").