Introduction à la Théorie de l'Information et au Codage

Marc Lelarge

10 avril 2012

Table des matières

1 Suites typiques et compression de données avec pertes **7**
- 1.1 Entropie . . . 7
- 1.2 Ensemble typique . . . 8
- 1.3 Codage de source avec perte . . . 9
- 1.4 Convexité et propriétés de l'entropie . . . 11
- 1.5 Entropie conditionnelle et Information mutuelle . . . 12
- 1.6 Test d'hypothèse . . . 13
- 1.7 Exercice : Définition axiomatique de l'entropie . . . 15

2 Codage pour des sources discrètes **17**
- 2.1 Mots code de longueur variable . . . 17
- 2.2 Un théorème de codage de source . . . 20
- 2.3 Un codage optimal : le codage de Huffman . . . 21
  - 2.3.1 Cas du code binaire : $D = 2$ . . . 21
  - 2.3.2 Extension au cas $D > 2$ . . . 23
- 2.4 Exercice : un test pour les codes non-ambigus . . . 24

3 Codage de source universel **27**
- 3.1 Codage universel pour une suite binaire . . . 27
- 3.2 Codage par automates à états finis . . . 29
- 3.3 Algorithme de Lempel-Ziv . . . 32
- 3.4 Exercice : optimalité de l'algorithme de Lempel-Ziv pour une source sans mémoire . . . 34

4 Propriétés de l'entropie et de l'information mutuelle **37**
- 4.1 Rappels . . . 37
- 4.2 Règles de la chaîne . . . 38
- 4.3 Inégalités de convexité . . . 39
  - 4.3.1 Inégalité logsum et applications . . . 41
- 4.4 Data Processing inequality . . . 42
- 4.5 Inégalité de Fano . . . 43
- 4.6 Exercice : Lien entre théorie de l'information et courses de chevaux . . . 44

5 Canaux discrets sans mémoire et leurs fonctions capacité-coût **47**
5.1 Définitions et codes sans erreur 47
5.2 Inégalité de Fano et réciproque du théorème de codage de canal 49
5.3 La fonction capacité-coût 50
5.4 Réciproque du théorème de codage de canal avec coût 53
5.5 Le théorème de codage de canal 54
5.6 Exercice : Canal avec feedback 58

6 Sources sans mémoire et leurs fonctions taux-distorsion **61**
6.1 La fonction taux-distorsion 61
6.2 Théorème de codage de source de Shannon 64

7 Le théorème de codage source-canal **67**

8 Codes linéaires **71**
8.1 Décodage par maximum de vraisemblance 71
8.2 Géométrie de Hamming 72
8.3 Codes linéaires 73
8.4 Codes de Hamming (binaires) 74
8.5 Décodage du syndrome 74

9 Codes cycliques **77**
9.1 Propriétés générales des codes cycliques 77
9.2 Classification des codes binaires cycliques de longueur 7 81

Notations

Pour des variables aléatoires (v.a.) discrètes X et Y à valeurs dans $\mathcal{X}$ et $\mathcal{Y}$ resp., on utilisera les notations suivantes pour $x \in \mathcal{X}$ et $y \in \mathcal{Y}$:

$$\begin{aligned} p(x) &= P(X = x) \\ p(y) &= P(Y = y) \\ p(x, y) &= P(X = x, Y = y) \\ p(x|y) &= P(X = x|Y = y) = p(x, y)/p(y). \end{aligned}$$

Lorsque ces notations sont ambigues, on pourra écrire $p_X(x), p_Y(y), p_{X,Y}(x, y), p_{X|Y}(x|y)$.

Chapitre 1

Suites typiques et compression de données avec pertes

1.1 Entropie

Une source (discrète) émet une suite de v.a. $\{U_i\}_{i=1}^{\infty}$ à valeurs dans un ensemble fini $\mathcal{U}$ appelé l'alphabet de la source. Si les U_i sont indépendants et identiquement distribués (i.i.d.) de loi P, la source est dite sans mémoire de distribution P.

Définition 1.1.1 *Soit U une variable aléatoire à valeurs dans un ensemble fini $\mathcal{U}$, de distribution de probabilité :*

$$p(u) = P(U = u),\ u \in \mathcal{U}\,.$$

Son entropie est, par définition, la quantité

$$H(U) = -E\left[\log(p(U))\right] = -\sum_{u\in\mathcal{U}} p(u)\log p(u)\,,$$

avec la convention $0\log 0 = 0$.

Le choix de la base du logarithme correspond à un choix d'unité. Sauf mention du contraire, on choisit par défaut la base 2. L'entropie s'exprime alors en bits.

1.2 Ensemble typique

Définition 1.2.1 *Pour $n \in \mathbb{N}$ et $\delta > 0$, l'ensemble typique A_δ^n par rapport à la distribution $p(u)$ est l'ensemble des suites $(u_1, \ldots, u_n) \in \mathcal{U}^n$ telles que :*

$$2^{-n(H(U)+\delta)} \leq p(u_1, \ldots, u_n) \leq 2^{-n(H(U)-\delta)}.$$

Théorème 1.2.1 *Pour tout $n \in \mathbb{N}$, $\delta > 0$, on a :*

1. *Si $(u_1, \ldots, u_n) \in A_\delta^{(n)}$ alors*

$$H(U) - \delta \leq -\frac{1}{n} \log p(u_1, \ldots, u_n) \leq H(U) + \delta.$$

2. *Pour tout $\epsilon > 0$ et pour n suffisament grand, on a :*

$$P\left(A_\delta^{(n)}\right) = P\left((U_1, \ldots, U_n) \in A_\delta^{(n)}\right) \geq 1 - \epsilon.$$

3. *le cardinal de l'ensemble $A_\delta^{(n)}$ est borné par :*

$$\left|A_\delta^{(n)}\right| \leq 2^{n(H(U)+\delta)},$$

et pour tout $\epsilon > 0$, pour n suffisament grand, ce cardinal est minoré par :

$$\left|A_\delta^{(n)}\right| \geq (1-\epsilon)2^{n(H(U)-\delta)}.$$

Remarque 1.2.1 *Le point 3. du Théorèm entraine directement :*

$$H(U) - \delta \leq \liminf_{n\to\infty} \frac{\log\left|A_\delta^{(n)}\right|}{n} \leq \limsup_{n\to\infty} \frac{\log\left|A_\delta^{(n)}\right|}{n} \leq H(U) + \delta.$$

Démonstration. Le point 1 est une application directe de la définition. Le point 2 découle de la loi faible des grands nombres en écrivant :

$$-\frac{1}{n} \log p\left(U_1, U_2, \ldots, U_n\right) = -\frac{1}{n} \sum_{i=1}^{n} \log p(U_i),$$

qui est une somme de v.a. i.i.d. de moyenne $-E\left[\log p(U)\right] = H(U)$.

Pour la première partie du point 3, on écrit :

$$1 = \sum_{(u_1,\ldots,u_n)\in\mathcal{U}^n} p(u_1,\ldots,u_n) \geq \sum_{(u_1,\ldots,u_n)\in A_\delta^{(n)}} p(u_1,\ldots,u_n) \geq \left|A_\delta^{(n)}\right| 2^{-n(H(U)+\delta)},$$

où la dernière inégalité provient de la définition de l'ensemble typique.

Pour la seconde partie du point 3, on a grâce au point 2 pour n suffisament grand :

$$1-\epsilon \leq P\left(A_\delta^{(n)}\right) \leq \left|A_\delta^{(n)}\right| 2^{-n(H(U)-\delta)}.$$

□

1.3 Codage de source avec perte

Nous considérons dans ce chapitre une notion très générale de codage que nous préciserons dans le chapitre suivant. Un codage binaire est une paire de fonctions

$$f : \mathcal{U}^k \to \{0,1\}^n, \text{ et } \phi : \{0,1\}^n \to \mathcal{U}^k.$$

Pour une source donnée, la probabilité d'erreur du code (f,ϕ) est

$$e(f,\phi) := P(\phi(f(U^{(k)})) \neq U^{(k)}),$$

avec $U^{(k)} = (U_1,\ldots,U_k)$ les k premiers symboles émis par la source.

Le but est de trouver des codes avec un ratio n/k petit et une probabilité d'erreur petite. Plus précisément, pour tout k, soit $n(k,\epsilon)$ le plus petit entier n tel qu'il existe un (k,n)-code satisfaisant $e(f,\phi) \leq \epsilon$.

Théorème 1.3.1 *Pour une source discrète sans mémoire de distribution $P(U = u) = p(u)$, on a pour tout $\epsilon \in (0,1)$:*

$$\lim_{k\to\infty} \frac{n(k,\epsilon)}{k} = H(U) = -\sum_{u\in\mathcal{U}} p(u)\log p(u).$$

Démonstration. L'existence d'un (k,n)-code binaire avec $e(f,\phi) \leq \epsilon$ est équivalente à l'existence d'un ensemble $A \subset \mathcal{U}^k$ avec $P(A) \geq 1-\epsilon$ et $|A| \leq 2^n$. A est alors l'ensemble des suites $u^{(k)} \in \mathcal{U}^k$ reproduites de manière exacte, c.a.d. telles que $\phi(f(u^{(k)})) = u^{(k)}$.

Soit $s(k,\epsilon)$ la taille mimimale d'un ensemble $A \subset \mathcal{U}^k$ avec $P(A) \geq 1-\epsilon$, c.a.d

$$s(k,\epsilon) = \min\{|A|;\ P(A) \geq 1-\epsilon\}.$$

Pour prouver le théorème, il suffit de montrer que pour $\epsilon \in (0,1)$,

$$\lim \frac{\log s(k,\epsilon)}{k} = H(U). \tag{1.1}$$

Pour tout $\delta > 0$, en prenant $A = A_\delta^{(k)}$ l'ensemble typique pour la source $p(u)$, on a pour k suffisament grand $P\left(A_\delta^{(k)}\right) \geq 1-\epsilon$ et donc :

$$s(k,\epsilon) \leq |A_\delta^{(k)}| \leq 2^{k(H(U)+\delta)},$$

donc

$$\limsup_k \frac{\log s(k,\epsilon)}{k} \leq H(U). \tag{1.2}$$

Inversement, pour tout $A \subset \mathcal{U}^k$ avec $P(A) \geq 1-\epsilon > 0$, le point 2 du Théorème 1.2.1 implique que pour k suffisament grand $P\left(A_\delta^{(k)}\right) \geq \frac{1+\epsilon}{2}$ et donc

$$P(A \cap A_\delta^{(k)}) \geq P(A) - (1 - P\left(A_\delta^{(k)}\right)) \geq \frac{1-\epsilon}{2}.$$

On a donc par définition de $A_\delta^{(k)}$,

$$|A| \geq |A \cap A_\delta^{(k)}| \geq \sum_{u^{(k)} \in A \cap A_\delta^{(k)}} p(u^{(k)}) 2^{k(H(U)-\delta)} \geq \frac{1-\epsilon}{2} 2^{k(H(U)-\delta)},$$

et donc pour tout $\delta > 0$,

$$\liminf_{k\to\infty} \frac{1}{k} \log s(k,\epsilon) \geq H(U) - \delta.$$

Ceci, avec (1.2), implique (1.1). □

Corollaire 1.3.1

$$0 \leq H(U) \leq \log |\mathcal{U}|$$

1.4 Convexité et propriétés de l'entropie

Lemme 1.4.1 *Si $(p_i : 1 \leq i \leq n)$ est une distribution de probabilité alors le minimum de la fonction*

$$G(q_1, \ldots, q_n) = -\sum p_i \log q_i,$$

sur toutes les distributions de probabilité $(q_1, \ldots, q_n)$ est atteint uniquement pour $q_k = p_k$, $1 \leq k \leq n$.

Démonstration. En utilisant l'inégalité de convexité $\log z \leq (z-1)\log e$ qui est une égalité uniquement lorsque $z = 1$, on obtient :

$$\log\left(\frac{q_k}{p_k}\right) \leq \left(\frac{q_k}{p_k} - 1\right)\log e,$$

avec égalité si et seulement si $q_k = p_k$. On a donc

$$G(p_1, \ldots, p_n) - G(q_1, \ldots, q_n) = \sum_k p_k \log\left(\frac{q_k}{p_k}\right) \leq \log e \sum_k (q_k - p_k) = 0.$$

□

On en déduit facilement les théorème suivants :

Théorème 1.4.1 *Pour tout n, $H(p_1, \ldots, p_n) \leq \log n$, avec égalité si et seulement si $p_1 = p_2 = \cdots = p_n = 1/n$.*

Démonstration. D'après le lemme précédent, on a :

$$H(p_1, \ldots, p_n) = G(p_1, \ldots, p_n) \leq G(1/n, \ldots, 1/n) = \log n,$$

avec égalité si et seulement si $p_i = 1/n$ pour tout $1 \leq i \leq n$. □

Théorème 1.4.2 *Si X et Y sont des v.a. (discrètes) alors $H(X,Y) \leq H(X) + H(Y)$, avec égalité si et seulement si X et Y sont indépendantes.*

Démonstration. On a

$$\begin{aligned} H(X) + H(Y) &= -\sum_x p(x)\log p(x) - \sum_y p(y)\log p(y) \\ &= -\sum_{x,y} p(x,y)\log p(x) - \sum_{x,y} p(x,y)\log p(y) \\ &= -\sum_{x,y} p(x,y)\log p(x)p(y). \end{aligned}$$

Donc par le Lemme 1.4.1, on a

$$H(X,Y) = -\sum_{x,y} p(x,y)\log p(x,y) \leq -\sum_{x,y} p(x,y)\log p(x)p(y) = H(X)+H(Y),$$

avec égalité si et seulement si $p(x,y) = p(x)p(y)$, c'est à dire si X et Y sont indépendantes. □

1.5 Entropie conditionnelle et Information mutuelle

Étant donné une v.a. X sur un espace de probabilité Ω et A un événement dans Ω, on définit l'entropie conditionnelle de X sachant A par

$$H(X|A) = -\sum_{k=1}^{m} P(X = x_k|A)\log P(X = x_k|A).$$

De la même manière si Y est une autre v.a., on définit l'entropie conditionnelle de X sachant Y par

$$\begin{aligned} H(X|Y) &= \sum_{j=1}^{m} H(X|Y = y_j)P(Y = y_j) \\ &= -\sum_{x,y} p(x,y)\log p(x|y). \end{aligned}$$

Il est facile de vérifier les propriétés suivantes :

$$\begin{aligned} H(X|X) &= 0 \\ H(X|Y) &= H(X) \text{ si } X \text{ et } Y \text{ sont indépendantes} \\ H(X|Y) &= 0 \text{ si et seulement si } X = g(Y) \text{ pour une fonction } g. \end{aligned}$$

Pour la dernière propriété, il suffit d'écrire $H(X|Y) = \sum H(X|Y = y_i)P(Y = y_i)$ donc pour que $H(X|Y) = 0$, il faut que $H(X|Y = y_i) = 0$ pour chaque i, c'est à dire qu'il existe x_i tel que $P(X = x_i|Y = y_i) = 1$. Donc X est déterminé par Y.

L'entropie d'une paire (X,Y) ne nécessite pas de nouvelle définition ! On notera $H((X,Y)) = H(X,Y)$. La différence $H(X,Y) - H(X)$ mesure la quantité d'information supplémentaire sur le couple (X,Y) donnée par Y si X est déjà connu. Comme montré dans le théorème suivant, cette différence est l'entropie conditionnelle de Y sachant X.

Théorème 1.5.1 *Pour toute paire de v.a. X, Y, on a $H(X,Y) = H(Y)+H(X|Y)$.*

Démonstration. On écrit

$$\begin{aligned} H(X,Y) &= -\sum_{x,y} p(x,y)\log p(x,y) \\ &= -\sum_{x,y} p(x,y)\log p(y)p(x|y) \\ &= -\sum_{y} p(y)\log p(y) - \sum_{x,y} p(x,y)\log p(x|y), \end{aligned}$$

ce qui est l'égalité souhaitée. □

Corollaire 1.5.1 *Pour toute paire de v.a. X, Y, $H(X|Y) \leq H(X)$ avec égalité si et seulement si X et Y sont indépendantes.*

Démonstration. On a $H(X|Y) = H(X,Y) - H(Y)$ et $H(X,Y) \leq H(X) + H(Y)$ avec égalité si et seulement si X et Y sont indépendantes. Le résultat en découle. □

Définition 1.5.1 *L'information mutuelle entre X et Y est définie par*

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

L'information mutuelle entre X et Y correspond à la diminution d'incertitude sur Y causée par la connaissance de X, c'est à dire la quantité d'information sur Y contenue 'dans' X. Elle est symmétrique en X et Y.

1.6 Test d'hypothèse

Problème : décider entre deux distributions P et Q à partir d'un échantillon de taille k, c.a.d. le résultat de k tirages indépendants. Un test est défini par un ensemble $A \subset \mathcal{U}^k$: si l'échantillon $(U_1, \ldots, U_k)$ appartient à A alors le test retourne l'hypothèse P sinon Q.

On considère un scénario où les hypothèses ne sont pas symmétriques. On désire une probabilité d'erreur au plus ϵ si P est la vraie distribution, c.a.d. $P(A) \geq 1-\epsilon$. Le but est alors de minimiser la probabilité d'erreur si l'hypothèse Q est vraie, c.a.d

$$\beta(k,\epsilon) = \min\{Q(A),\ \text{t.q. } A \subset \mathcal{U}^k,\ P(A) \geq 1-\epsilon\}.$$

Théorème 1.6.1 *Pour tout $\epsilon \in (0,1)$, on a*

$$\lim_{k\to\infty} \frac{1}{k}\log\beta(k,\epsilon) = -\sum_{u\in\mathcal{U}} p(u)\log\frac{p(u)}{q(u)}.$$

Définition 1.6.1 *L'entropie relative ou distance de Kullback-Leibler entre deux distributions p et q est définie par :*

$$D(p\|q) = E_p\left[\log\frac{p(U)}{q(U)}\right] = \sum_{u\in\mathcal{U}} p(u)\log\frac{p(u)}{q(u)},$$

avec les conventions $0\log\frac{0}{0}=0$, $0\log\frac{0}{q}=0$ *et* $p\log\frac{p}{0}=\infty$.

Pour deux distributions p et q telles que $p(u)q(u) > 0$ pour tout $u \in \mathcal{U}$, on définit l'ensemble $A_\delta^n(p\|q)$ par l'ensemble des suites $(u_1,\ldots,u_n)\in\mathcal{U}^n$ telles que :

$$2^{-n(D(p\|q)+\delta)} \le \frac{q(u_1,\ldots,u_n)}{p(u_1,\ldots,u_n)} \le 2^{-n(D(p\|q)-\delta)}.$$

Théorème 1.6.2 *Si $p(u)q(u) > 0$ pour tout $u\in\mathcal{U}$, pour tout $n\in\mathbb{N}$ et $\delta > 0$, on a :*

1. *Si $(u_1,\ldots,u_n)\in A_\delta^{(n)}(p\|q)$ alors*

$$D(p\|q) - \delta \le \frac{1}{n}\log\frac{p(u_1,\ldots,u_n)}{q(u_1,\ldots,u_n)} \le D(p\|q) + \delta.$$

2. *Pour tout $\epsilon > 0$ et pour n suffisament grand, on a :*

$$P\left(A_\delta^{(n)}(p\|q)\right) \ge 1-\epsilon.$$

3. *Pour tout $\epsilon > 0$ et pour n suffisament grand, on a :*

$$(1-\epsilon)2^{-n(D(p\|q)+\delta)} \le Q\left(A_\delta^{(n)}(p\|q)\right) \le 2^{-n(D(p\|q)-\delta)}.$$

Démonstration. La démonstration est similaire à celle pour l'ensemble typique. Le premier point découle de la définition et le point 2 de la loi faible des grands nombres. Pour le point 3, on écrit :

$$\begin{aligned} Q\left(A_\delta^{(n)}(p\|q)\right) &= \sum_{(u_1,\ldots,u_n)\in A_\delta^{(n)}(p\|q)} q(u_1,\ldots,u_n) \\ &\le \sum_{(u_1,\ldots,u_n)\in A_\delta^{(n)}(p\|q)} p(u_1,\ldots,u_n)2^{-n(D(p\|q)-\delta)} \le 2^{-n(D(p\|q)-\delta)}, \end{aligned}$$

et pour l'autre borne :

$$Q\left(A_\delta^{(n)}(p\|q)\right) \ge \sum_{(u_1,\ldots,u_n)\in A_\delta^{(n)}(p\|q)} p(u_1,\ldots,u_n)2^{-n(D(p\|q)+\delta)} \ge (1-\epsilon)2^{-n(D(p\|q)+\delta)},$$

où la dernière inégalité découle du point 2. □

Intuitivement, l'ensemble $A_\delta^{(n)}(p\|q)$ permet de distinguer les distributions p et q : si la suite est tirée selon P, alors elle appartient avec grande probabilité à $A_\delta^{(n)}(p\|q)$ tandis que si elle est tirée selon Q, la probabilité d'appartenance à $A_\delta^{(n)}(p\|q)$ décroît exponentiellement vite vers zéro.

Démonstration.(du Théorème 1.6.1)
Le cas $p(u)q(u) > 0$ pour tout $u \in \mathcal{U}$ découle directement du théorème précédent. En effet, on a directement,

$$\frac{1}{k}\log s(k,\epsilon) \leq \frac{1}{k}\log Q\left(A_\delta^{(n)}(p\|q)\right) \leq -D(p\|q) + \delta.$$

Inversement, pour tout $A \subset \mathcal{U}^k$ avec $P(A) \geq 1-\epsilon$, on a $P\left(A \cap A_\delta^{(k)}(p\|q)\right) \geq 1-2\epsilon$ pour k suffisament grand et donc

$$Q(A) \geq Q\left(A \cap A_\delta^{(k)}(p\|q)\right) \geq (1-2\epsilon)2^{-k(D(p\|q)+\delta)}.$$

On a donc $\frac{1}{k}\log s(k,\epsilon) \geq \frac{1}{k}\log(1-2\epsilon) - D(p\|q) - \delta$. □

1.7 Exercice : Définition axiomatique de l'entropie

Étant donné une distribution de probabilité $p_1, \ldots, p_n$, on cherche une fonction $H(p_1, \ldots, p_n)$ quantifiant "l'incertitude" associée à cette distribution. On postule les conditions suivantes pour la fonction H :

(A1) $H(p_1, \ldots, p_n)$ est maximum pour $p_1 = p_2 = \cdots = p_n = 1/n$.

(A2) H est une fonction symétrique en ses arguments.

(A3) $H(p_1, \ldots, p_n) \geq 0$ avec égalité quand un des p_i vaut 1.

(A4) $H(p_1, \ldots, p_n, 0) = H(p_1, \ldots, p_n)$.

(A5) $H\left(\frac{1}{n}, \ldots, \frac{1}{n}\right) \leq H\left(\frac{1}{n+1}, \ldots, \frac{1}{n+1}\right)$.

(A6) la fonction H est continue.

(A7) pour des entiers m et n,

$$H\left(\frac{1}{mn}, \ldots, \frac{1}{mn}\right) = H\left(\frac{1}{m}, \ldots, \frac{1}{m}\right) + H\left(\frac{1}{n}, \ldots, \frac{1}{n}\right).$$

(A8) pour $p = p_1 + \cdots + p_m$ et $q = q_1 + \cdots + q_n$ où tous les p_i, q_i sont positifs. Si p et q sont strictement positifs tels que $p + q = 1$, on a :

$$H(p_1, \ldots, p_m, q_1, \ldots, q_n) = H(p,q) + pH(p_1/p, \ldots, p_m/p) + qH(q_1/q, \ldots, q_n/q).$$

1. Jusifier les différents axiomes.
2. Montrer que la fonction $g(n) = H\left(\frac{1}{n}, \ldots, \frac{1}{n}\right)$ est de la forme $g(n) = A \ln n$ pour un certain A.
3. En déduire la forme de $H(p, 1-p)$ quand p est rationel.
4. Conclure.

Chapitre 2

Codage pour des sources discrètes

2.1 Mots code de longueur variable

Définition 2.1.1 *Un code de source C pour une variable aléatoire $U \in \mathcal{U}$ est une fonction de $\mathcal{U}$ vers $\mathcal{D}^*$ (l'ensemble des mots finis sur un alphabet D-aire). $C(u)$ est le mot code correspondant à u, et $l(u)$ sa longueur.*

Définition 2.1.2 *La longueur moyenne $L(C)$ d'un code pour la variable aléatoire U de distribution de probabilité $p(u)$ est :*

$$L(C) = \sum_{u \in \mathcal{U}} p(u) l(u)$$

Remarque 2.1.1 *Dans la suite, on supposera sans perte de généralité : $\mathcal{D} = \{0, 1, \ldots, d-1\}$.*

Exemple 2.1.1: Sur un alphabet binaire, si :

$$\begin{array}{ll} P(U=1) = 1/2 & C(1) = 0 \\ P(U=2) = 1/4 & C(2) = 10 \\ P(U=3) = 1/8 & C(3) = 110 \\ P(U=4) = 1/8 & C(4) = 111 \end{array}$$

on a $H(U) = 1.75$ bits et $L(C) = 1.75$ bits. Par exemple : 0110111100110 se décode en 134213.

Définition 2.1.3 *Un code C est dit* non-ambigu *si :*

$$x \neq y \Rightarrow C(x) \neq C(y).$$

Définition 2.1.4 *L'extension C^* du code C est la fonction des mots finis de $\mathcal{U}$ vers les mots finis de $\mathcal{D}$ définie par*

$$C(u_1 \dots u_n) := C(u_1) \dots C(u_n) \text{ (concaténation)}$$

Définition 2.1.5 *Un code C est dit* uniquement décodable *si son extension C^* est non-ambigüe.*

Définition 2.1.6 *Un code est dit* instantané *si aucun mot code n'est le préfixe d'un autre mot code.*

EXEMPLE 2.1.2: Parmi les codes suivants :

U	code 1	code 2	code 3
1	0	10	0
2	010	00	10
3	01	11	110
4	10	110	111

Le code 3 est instantané.
Le code 2 est uniquement décodable (il suffit de regarder la parité du nombre de 0 après 11).
Le code 1 est non-ambigu, mais non uniquement décodable, par exemple 010 peut se décoder par $2, 14$ ou 31.

Un code instantané est uniquement décodable. De plus un code instantané peut être décodé sans référence aux mots code future puisque la fin d'un mot code est reconnaissable immédiatement.

Théorème 2.1.1 (Inégalité de Kraft) *Pour un code instantané sur un alphabet de taille D, les longueurs des mots code $l_1, \dots, l_m$ doivent vérifier :*

$$\sum_i D^{-l_i} \leq 1$$

Inversement, étant donné une suite de longueurs vérifiant cette inégalité, il existe un code instantané avec des mots code ayant ces longueurs.

Démonstration. Pour prouver le premier point, on peut considérer l'arbre de codage du code C.

Soit l_{max} la longueur du plus long mot code. Un mot code de longueur l_i a $D^{l_{max}-l_i}$ descendants à la profondeur l_{max} qui doivent être disjoints des descendants des autres mots code par la propriété du préfixe. On a donc :

$$\sum_i D^{l_{max}-l_i} \leq D^{l_{max}}.$$

Inversement étant donné des longueurs $\ell_1, \ldots, \ell_m$ satisfaisant l'inégalité de Kraft, on peut toujours construire un arbre de codage comme précédemment : dans l'arbre D-aire, associer au premier (pour l'ordre lexicographique de l'arbre) noeud de profondeur l_1 le mot code 1 et retirer ses descendants. Associer alors au premier noeud restant de profondeur l_2 le mot code 2 et ainsi de suite. □

Théorème 2.1.2 (McMillan) *Les longueurs des mots code d'un code D-aire uniquement décodable doivent satisfaire l'inégalité de Kraft.*

Une conséquence immédiate est que les codes instantanés seront tout aussi performants (pour ce qui concerne leurs longueurs) que les codes uniquement décodables (et pas moins, comme on aurait pu le penser).

Démonstration. Soit k un entier.

On a :

$$\begin{aligned}\left(\sum_{u\in\mathcal{U}} D^{-l(u)}\right)^k &= \sum_{u_1\in\mathcal{U}} \cdots \sum_{u_k\in\mathcal{U}} D^{-l(u_1)-l(u_2)\cdots-l(u_k)} \\ &= \sum_{(u_1,\ldots,u_k)\in\mathcal{U}^k} D^{-l(u_1\ldots u_k)} \\ &= \sum_{m=1}^{kl_{max}} A(m) D^{-m}\end{aligned}$$

où :

$$A(m) = |\{(u_1 \ldots u_k) \in \mathcal{U}^k, l(u_1 \ldots u_k) = m\}|$$

On a $A(m) \leq D^m$ car le code est uniquement décodable, et donc :

$$\sum_{u\in\mathcal{U}} D^{-l(u)} \leq (kl_{max})^{1/k}$$

Or $(kl_{max})^{1/k} \xrightarrow[k\to\infty]{} 1$, ce qui conclut la preuve □

2.2 Un théorème de codage de source

Théorème 2.2.1 *Etant donné une source discrète à valeurs dans $\mathcal{U}$ et d'entropie $H(U)$, et étant donné un alphabet de D symboles pour le code, il est possible de coder chaque lettre de la source de manière instantanée et telle que la longueur moyenne des mots satisfasse : $L(C) < \frac{H(U)}{\log D} + 1$.*

De plus, pour tout code uniquement décodable : $L(C) \geq H(U)/\log D$.

Démonstration. On a :

$$\begin{aligned} H(U) - L(C)\log D &= \sum_u p(u)\log\frac{1}{p(u)} - \sum_u p(u)l(u)\log D \\ &= \sum_u p(u)\log\frac{D^{-l(u)}}{p(u)} \end{aligned}$$

On sait par ailleurs que pour $z > 0$, $\log z \leq (z-1)\log e$, on a donc :

$$\begin{aligned} H(U) - L(C) &\leq (\log e)\big(\sum_u D^{-l(u)} - \underbrace{\sum_u p(u)}_{=1}\big) \\ &\leq 0\text{par (Mcmillan)} \end{aligned}$$

Pour l'autre inégalité, on choisit $l(u)$ tel que $D^{-l(u)} \leq p(u) < D^{-l(u)+1}$.
On a donc :

$$\sum_u D^{-l(u)} \leq 1$$

D'après l'inégalité de Kraft, il existe donc un code instantané avec ces longueurs, de plus

$$\begin{aligned} \log p(u) &< (-l(u)+1)\log D \\ l(u) &< \frac{-\log p(u)}{\log D} + 1 \end{aligned}$$

et

$$L(C) = \sum p(u)l(u) < \frac{H(U}{\log D} + 1$$

□

Théorème 2.2.2 *Pour une source discrète sans mémoire d'entropie $H(U)$ et un alphabet à D symboles, il est possile de coder les suites de k lettres de la source de sorte que :*

1. *La propriété du préfixe soit satisfaite*
2. *La longueur moyenne des mots code par lettre source vérifie :*

$$H(U)/\log D \leq L^k/k < H(U)/\log D + 1/k$$

où $L^k = \sum_{u_1 \dots u_k} l(u_1 \dots u_k) p(u_1 \dots u_k)$

Démonstration. Il suffit de vérifier que $H(U^{(k)}) = kH(U)$, et le résultat découle du théorème précédent.

On a bien $H(U^{(k)}) = \sum_{u_1 \dots u_k} p(u_1 \dots u_k) \log p(u_1 \dots u_k)$ et par la propriété sans mémoire de la source, $p(u_1 \dots u_k) = p(u_1) \dots p(u_k)$, d'où le résultat.

□

2.3 Un codage optimal : le codage de Huffman

On présente ici le codage de Huffman. Il est optimal en ce sens qu'il n'existe pas de code uniquement décodable avec une longueur moyenne inférieure.

Dans toute la suite, on se restreint aux codes instantnés sans perte de généralité par l'inégalité de McMillan.

2.3.1 Cas du code binaire : $D = 2$

On suppose que $\mathcal{U} = \{U_1, \dots, U_k\}$, avec $p(u_1) \geq p(u_2) \geq \dots \geq p(u_k)$.

Lemme 2.3.1 *Pour tout $k \geq 2$, un code binaire optimal existe pour lequel les mots code les moins probables $C(u_k)$ et $C(u_{k-1})$ ont la même longueur et diffèrent par le dernier bit. (disons que $C(u_k)$ finit par 1 et $C(u_{k-1})$ par 0)*

Démonstration. Si $l(u_k) < \max_i l(u_i) := l(u_j)$ alors on obtient un meilleur code en interchangeant les mots codant u_k et u_j. En effet, on change $L(C)$ de :

$$\begin{aligned} \Delta &= p(u_j)l(u_k) + p(u_k)l(u_j) - p(u_k)l(u_k) - p(u_j)l(u_j) \\ &= (p(u_j) - p(u_k))(l(u_k) - l(u_j)) \leq 0. \end{aligned}$$

On peut donc supposer $l(u_k) = \max_i l(u_i)$.

Si maintenant $C(u_k)$ est le seul mot de longueur $l(u_k)$, on obtient un meilleur code instantané en tronquant les derniers bits de $C(u_k)$. Donc il existe u_i tel que $l(u_i) = l(x_k)$ et $C(u_i)$ et $C(u_k)$ diffèrent dans le dernier digit. Donc $l(u_i) \geq l(u_{k-1})$ et par le même argument que précédemment, interchanger $C(u_i)$ et $C(u_{k-1})$ n'augmente pas $L(C)$. □

On a donc réduit le problème de construction d'un code optimal à celui de construire $C(u_1), \ldots, C(u_{k-2})$ et trouver les $l(u_k) - 1$ premiers digits de $C(u_k)$.

On définit maintenant l'ensemble réduit : $\mathcal{U}' = \{u'_1, \ldots, u'_{k-1}\}$ avec la v.a U' associée : $p(u'_j) = p(u_j)$ si $j \leq k-2$ et $p(u'_{k-1}) = p(u_k) + p(u_{k-1})$.

Il y a une bijection entre les codes instantanés pour U' et les codes instantanés pour U pour lesquels $C(u_k)$ et $C(u_{k-1})$ ne diffrent que par le dernier digit, $C(u_k)$ finissant par un 1 et $C(u_{k-1})$ par un 0.

Lemme 2.3.2 *Si un code instantané est optimal pour U', le code instantané correspondant pour U est optimal.*

Démonstration.

$$l(u_j) = \begin{cases} l(u'_j) & \text{si } j \leq k-2 \\ l(u'_{k-1}) + 1 & \text{si } j \geq k-1 \end{cases}$$

Donc,

$$\begin{aligned} L(C) &= \sum p(u_j) l(u_j) \\ &= \sum_{j \leq k-2} p(u'_j) l(u'_j) + (p(u_{k-1}) + p(u_k))(l(u'_{k-1}) + 1) \end{aligned}$$

Or $p(u_{k-1}) + p(u_k) = p(u'_{k-1})$, donc :

$$L(C) = L(C') + p(u'_{k-1}).$$

Comme $p(u'_{k-1})$ ne dépend pas de C', on peut minimiser $L(C)$ sur la classe des codes où $C(u_k)$ et $C(u_{k-1})$ ne diffèrent que sur le dernier digit en minimiasant $L(C')$. Par le lemme 2.3.1, un tel code minimise $L(C)$ sur tous les codes instantanés. □

Application On construit donc l'arbre de codage de Huffman de proche en proche en rassemblant à chaque étape les deux noeuds de plus faible probabilité et en affectant la somme de ces probabilités au noeud père.

Voici un exemple :

mot code	message	$p(u_k)$
00	u_1	0.3
01	u_2	0.25
10	u_3	0.25
110	u_4	0.1
111	u_5	0.1

2.3.2 Extension au cas $D > 2$

On définit un arbre de codage *complet* comme un arbre de codage pour lequel tous les noeuds intermédiaires ont D enfants.

Lemme 2.3.3 *Le nombre de feuilles dans un arbre de codage complet est de la forme $D + m(D-1)$ pour un certain entier m.*

Démonstration. Le plus petit arbre complet a D feuilles, le second plus petit en a D-1+D (on remplace une feuille de l'arbre précédent par un noeud à D enfants), d'où le résultat par récurrence. □

Pour un code instantané, nous complétons son arbre de codage en rajoutant B feuilles (non utilisées par le code).

Pour un code optimal, toutes les feuilles non utilisées doivent être au même niveau que le mot code le plus long, et ne diffèrent que par le dernier digit.

Un code optimal doit donc avoir au plus $D-2$ feuilles inutilisées.

Si K le nombre de mots code et B le nombre de feuilles inutilisées, on doit avoir :

$$B + K \quad = \quad m(D-1) + D \text{ et } B \leq D-2,$$

donc $K - 2 = m(D-1) + (D-2-B)$ et $0 \leq D-2-B \leq D-2$

ainsi, $B = D - 2 - ((K-2) mod (D-1))$.

En suivant le Lemme 2.3.1, un code optimal existe pour lequel les B feuilles inutilisées et les $D-B$ mots code les moins probables diffèrent par le dernier digit. Donc la première étape consiste à grouper les $D-B$ noeuds les moins probables. Ensuite à chaque itération, l'ensemble réduit est de cardinal $D + m(D-1)$ et on regroupe les D noeuds les moins probables.

EXEMPLE 2.3.1: Pour $D = 3$ et $K = 6$, il faut rajouté 1 feuille inutilisée et on obtient dans cet exemple :

mot code	message	$p(u_k)$
0	u_1	0.4
1	u_2	0.3
20	u_3	0.2
21	u_4	0.05
220	u_5	0.03
221	u_6	0.02

2.4 Exercice : un test pour les codes non-ambigus

Le but de cet exercice est de donner un algorithme qui permet de vérifier si un code est non-ambigu. Voici un exemple d'un code binaire ambigu :

$$C = \{1, 011, 01110, 1110, 10011\}. \tag{2.1}$$

Le mot $w = 011101110011$ a deux factorisations :

$$w = (01110)(1110)(011) = (011)(1)(011)(10011).$$

L'alphabet D-aire est noté $\mathcal{D}$. L'ensemble des mots sur $\mathcal{D}$ est noté $\mathcal{D}^*$. Pour $x, y \in \mathcal{D}^*$, on définit :

$$x^{-1}y = \{z \in \mathcal{D}^*;\ xz = y\} \text{ et, } xy^{-1} = \{z \in \mathcal{D}^*;\ x = zy\}.$$

Pour des ensembles X, Y de $\mathcal{D}^*$, on étend ces définitions comme suit :

$$X^{-1}Y = \cup_{x\in X} \cup_{y\in Y} x^{-1}y \text{ et, } XY^{-1} = \cup_{x\in X} \cup_{y\in Y} xy^{-1}.$$

Les puissances de X sont définies par $X^0 = \{e\}$ où e est le mot vide, $X^1 = X$ et $X^{n+1} = XX^n = \{xy,\ x \in X, y \in X^n\}$, pour $n \geq 1$.

On voit un code D-aire C comme un sous-ensemble de $\mathcal{D}^+ = \mathcal{D}^* - e$. On définit alors

$$\begin{aligned} U_1 &= C^{-1}C - e, \\ U_{n+1} &= C^{-1}U_n \cup U_n^{-1}C, \text{ pour } n \geq 1. \end{aligned}$$

Nous allons montrer le théorème suivant :

Théorème 2.4.1 *Le code $C \subset \mathcal{D}^+$ est un code non-ambigu si et seulement si aucun des ensembles U_n définis ci-dessus ne contient le mot vide.*

1) Ecrire U_1, U_2, U_3 pour le code binaire donné par (2.1). Que vaut U_1 pour un code instantané ? Que valent les U_n pour l'exemple de code binaire vu en cours : $\{10, 00, 11, 110\}$?
2) Montrer par induction sur k que : pour tout $n \geq 1$ et $k \in \{1, \ldots, n\}$, on a $e \in U_n$ ssi il existe un mot $u \in U_k$ et des entiers $i, j \geq 0$ tels que :

$$uC^i \cap C^j \neq \emptyset \text{ et } i + j + k = n. \tag{2.2}$$

3) En déduire le Théorème 2.4.1.

1) Pour le code donné par (2.1), on a :

$$\begin{aligned} U_1 &= \{10, 110, 0011\}, \quad C^{-1}U_1 = \{0, 10\}, \quad U_1^{-1}C = \{011\}; \\ U_2 &= \{0, 10, 011\}, \quad C^{-1}U_2 = \{0, e\}, \quad U_2^{-1}C = \{11, 110, 011, e, 10\}. \end{aligned}$$

donc $e \in U_3$ et C est ambigu.
Pour un code instantané, on a $U_1 = \emptyset$.
Pour l'exemple vu en cours, on a $U_1 = U_n = \{0\}$.

2) On fait une induction décroissante sur k. Pour $k = n$: si $e \in U_n$, il suffit de prendre $u = e$, $i = j = 0$. Inversement si (2.2) est vérifiée pour $k = n$, on a $i = j = 0$ et donc $u = e$.
Soit $n > k \geq 1$, on suppose que l'equivalence est vraie pour $n, n-1, \ldots, k+1$.
Si $e \in U_n$ alos par induction, il existe $v \in U_{k+1}$ tel que $vx = y$ avec $x \in C^i$ et $y \in C^j$ et $i + j + k + 1 = n$. Par définition de U_{k+1}, on a
– soit $zv = u$ avec $z \in C$ et $u \in U_k$. Dans ce cas, $ux = zvx = zy$ avec $x \in C^i$ et $zy \in C^{j+1}$ donc $uC^i \cap C^{j+1} \neq \emptyset$.
– soit $z = uv$ avec $z \in C$ et $u \in U_k$. Dans ce cas, $zx = uvx = uy$ avec $zx \in C^{i+1}$ et $y \in C^j$ donc $C^{i+1} \cap uC^j \neq \emptyset$.
Dans les deux cas (2.2) est satisfaite.
Inversement, supposons qu'il existe $u \in U_k$ et $i, j \geq 0$ avec

$$uC^i \cap C^j \neq \emptyset,\ i + j + k = n.$$

On peut donc écrire : $ux_1x_2 \ldots x_i = y_1y_2 \ldots y_j$. Si $j = 0$ alors $i = 0$ et $k = n$.
Pour $j \geq 1$, on distingue à nouveau deux cas selon les longueurs respectives de u et y_1 :
– si $u = y_1v$ pour un $v \in \mathcal{D}^+$, alors $v \in C^{-1}U_k \subset U_{k+1}$ et de plus $vx_1x_2 \ldots x_i = y_2 \ldots y_j$. Donc $vC^i \cap C^{j-1} \neq \emptyset$ et par l'hypothèse d'induction $e \in U_n$.
– si $y_1 = uv$ pour un $v \in \mathcal{D}^+$, alors $v \in U_k^{-1}C \subset U_{k+1}$ et $x_1x_2 \ldots x_i = vy_2 \ldots y_j$. Donc $C^i \cap uC^{j-1} \neq \emptyset$ et $e \in U_n$.

3) Si C est ambigu, alors il existe une relation :

$$x_1x_2 \ldots x_p = y_1y_2 \ldots y_q, \quad x_1 \neq y_1.$$

On peut supposer sans perte de généralité que $x_1 = y_1u$ pour un $u \in \mathcal{D}^+$.
On a alors $u \in U_1$ et $uC^{p-1} \cap C^{q-1} \neq \emptyset$, d'où $e \in U_{p+q-1}$.
Inversement si $e \in U_n$. Prenons $k = 1$ dans la question précédente : il existe $u \in U_1$ et des entiers $i, j \geq 0$ tels que $uC^i \cap C^j \neq \emptyset$. Comme $u \in U_1$, on a $xu = y$ pour $x, y \in C$ et $x \neq y$ car $u \neq e$. Il découle de $xuC^i \cap xC^j \neq \emptyset$ que $yC^i \cap xC^j \neq \emptyset$ montrant que C est ambigu.

Chapitre 3

Codage de source universel

On cherche maintenant à trouver un codage pour une une suite quelconque $u^{(n)}$ sans faire aucune hypothèse probabiliste. Nous commençons par voir un exemple simple d'un tel codage pour une suite binaire, avant d'étudier l'algorithme de Lempel-Ziv.

3.1 Codage universel pour une suite binaire

Pour coder une suite binaire $u^{(n)} \in \{0,1\}^n$, on étudie l'algorithme offline[1] suivant :
- envoyer le nombre de 1 dans la suite : $\sum_{i=0}^{n} u_i$, en $\lceil \log_2(n+1) \rceil$ bits ;
- envoyer l'indice de la suite parmi toutes les suites ayant k 1, en $\lceil \log \binom{n}{k} \rceil$ bits.

Il faut donc au total :

$$\ell(u^{(n)}) \leq \log(n+1) + \log \binom{n}{k} + 2$$

On relie cette valeur à l'entropie grâce au lemme suivant.

Lemme 3.1.1 *Pour $k \neq 0, n$, on a*

$$\sqrt{\frac{n}{8k(n-k)}} \leq \binom{n}{k} 2^{-nH(\frac{k}{n})} \leq \sqrt{\frac{n}{\pi k(n-k)}}$$

1. un algorithme qui doit lire l'ensemble du message avant de pouvoir commencer le codage

Démonstration. On rappelle la formule de Stirling :

$$\sqrt{2\pi n}\left(\frac{n}{e}\right)^n \leq n! \leq \sqrt{2\pi n}\left(\frac{n}{e}\right)^n \exp\left(\frac{1}{12n}\right)$$

On note alors $k = np$ et $q = 1-p$. On rappelle la notation $H(p) = -p\log(p) - q\log(q)$, de telle sorte que $2^{-nH(p)} = p^{np}q^{nq}$.

On a alors,

$$\begin{aligned}\binom{n}{k} = \binom{n}{np} &\leq \frac{\sqrt{2\pi n}\left(\frac{n}{e}\right)^n \exp\left(\frac{1}{12n}\right)}{\sqrt{2\pi np}\left(\frac{np}{e}\right)^{np}\sqrt{2\pi nq}\left(\frac{nq}{e}\right)^{nq}} \\ &= \frac{1}{\sqrt{2\pi npq}} \times \frac{1}{p^{np}q^{nq}} \exp\left(\frac{1}{12n}\right) \\ &< \frac{1}{\sqrt{\pi np}} 2^{nH(p)},\end{aligned}$$

Car $\exp\left(\frac{1}{12n}\right) < \sqrt{2}$.

De même pour la borne inférieure, on a

$$\begin{aligned}\binom{n}{np} &\geq \frac{\sqrt{2\pi n}\left(\frac{n}{e}\right)^n}{\sqrt{2\pi np}\left(\frac{np}{e}\right)^{np}\sqrt{2\pi nq}\left(\frac{nq}{e}\right)^{nq}} \exp\left(-\frac{1}{12np} - \frac{1}{12nq}\right) \\ &= \frac{1}{\sqrt{2\pi npq}} 2^{nH(p)} \exp\left(-\frac{1}{12np} - \frac{1}{12nq}\right).\end{aligned}$$

On distingue alors plusieurs cas :
- si $np \geq 1$ et $nq \geq 3$, alors $\exp(-(\frac{1}{12np} + \frac{1}{12nq})) \geq \exp(-\frac{1}{9}) > \frac{\sqrt{\pi}}{2}$,
- si $np = 1$ et $nq = 1$ alors $n = 2$ et $p = \frac{1}{2}$, la borne vaut 2 et est correcte.
- si $np = 1$ et $nq = 2$ alors $n = 3$ et $p = \frac{1}{3}$, la borne vaut 2.92 et est correcte.
- si $np = 2$ et $nq = 2$ alors $n = 4$ et $p = \frac{1}{4}$, la borne vaut 5.66 et est correcte.

□

On a donc

$$\begin{aligned}\ell(u^{(n)}) &\leq \log(n+1) + nH\left(\frac{k}{n}\right) - \frac{1}{2}\log(n) - \frac{1}{2}\log(\pi\frac{k}{n}\frac{n-k}{n}) + 2 \\ &\leq nH\left(\frac{k}{n}\right) + \frac{1}{2}\log(n) - \frac{1}{2}\log(\pi\frac{k}{n}\frac{n-k}{n}) + 3\end{aligned}$$

Donc le coût pour décrire cette suite est de $\approx \frac{1}{2}\log(n)$ bits en plus du coût optimal de $nH\left(\frac{k}{n}\right)$ pour une distribution de Bernoulli correspondant à $p = \frac{k}{n}$.

3.2 Codage par automates à états finis

Nous allons dans un premier temps étudier la compression d'une suite infinie u par des automates finis. Nous établirons alors une borne sur le taux de compression de tels algorithmes, avant de montrer que l'algorithme de Lempel-Ziv atteint cette borne.

Dans toute la suite, chaque symbole de la source appartient à un alphabet fini ayant J symboles avec $J \geq 2$.

Définition 3.2.1 *Un automate à espace d'états fini est composé d'une tête de lecture se trouvant dans un certain état (parmi un ensemble fini). L'automate lit l'entrée symbole par symbole, chacun d'entre eux entraînant un changement d'état et l'émission d'un mot, éventuellement vide. Les changements sont régis par une table de transitions qui est une caractéristique de l'automate.*

Pour l'entrée $u = u_1u_2u_3\ldots$, l'automate produit $y = y_1y_2y_3,\ldots$ en visitant les états $z = z_1z_2z_3\ldots$ donnés par :

- $y_k = f(z_k, u_k)$ à valeurs $\{0,1\}^*$, pour $k \geq 1$,
- $z_{k+1} = g(z_k, u_k)$ à valeurs dans l'espace d'états (fini), pour $k \geq 1$.

Les fonctions f et g correspondent à une consultation de la table des transitions.

On notera $u_k^j = u_k u_{k+1}\ldots u_j$ et $f(z_k, u_k^j) = y_k^j$, $g(z_k, u_k^j) = z_{j+1}$.

Le décodeur a connaissance de l'automate et de son état initial. Il doit être capable de reconstruire u à partir de y.

Définition 3.2.2 *On dit qu'un encodeur est sans perte d'information, ou SPI, si $u_r^s \neq v_r^t$ alors pour tout z_r on a :*

- *soit $f(z_r, u_r^s) \neq f(z_r, v_r^t)$,*
- *soit $g(z_r, u_r^s) \neq g(z_r, v_r^t)$.*

Pour un encodeur n'étant pas SPI, il est impossible de retrouver u à partir de y. Notons cependant qu'un encodeur SPI n'est pas nécessairement "uniquement décodable", comme le montre l'exemple 3.2.

Nous calculons maintenant une borne inférieure sur le nombre de bits utilisés par symbole d'entrée pour tout encodeur SPI. Cette borne s'appliquera également aux encodeurs SPI conçus en connaissant U à l'avance, comme le fait l'algorithme de Huffman.

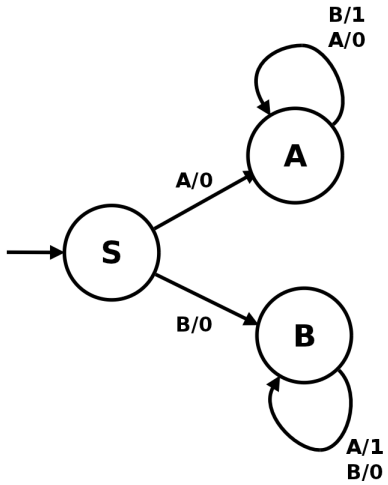

FIGURE 3.1 – Automate d'encodage. Les suites infinies $AAAA\ldots$ et $BBBB\ldots$ sont codées par $0000\ldots$ et donc indistinguables pourtant l'automate est SPI.

Définition 3.2.3 *Pour un encodeur E, son ratio de compression pour u est défini par*

$$\rho_E(u_1^n) = \frac{1}{n}\ell\left(y_1^n\right).$$

On définit alors :

$$\rho_s(u_1^n) = \min\{\rho_E(u_1^n), E \text{ encodeur SPI à } s \text{ états}\}$$

On note que $\rho_s(u_1^n) \leq \lceil \log J \rceil$.

On définit alors la compressibilité de u, notée $\rho(u)$ par :

$$\begin{aligned} \rho_s(u) &= \limsup_{n\to\infty} \rho_s(u_1^n) \\ \rho(u) &= \lim_{s\to\infty} \downarrow \rho_s(u). \end{aligned}$$

Soit $c(u_1^n)$ le nombre maximum de mots distincts en lesquels u_1^n peut être découpé (le mot vide ε inclus). On a donc $c(u_1^n) \geq 1$.

Si u_1^n est découpé en $c \geq 1$ mots distincts, on définit m et $0 \leq r \leq J^m$ tels que

$$c = \sum_{k=0}^{m-1} J^k + r.$$

Si un tel c est donné, le n minimal est obtenu par $ngeq \sum_{k=0}^{m-1} kJ^k + mr$ car il y a J^k mots de longueur k.

Pour $J \geq 2$, on a

$$\sum_{k=0}^{m-1} J^k = \frac{J^m - 1}{J - 1}$$

$$\sum_{k=0}^{m-1} kJ^k = m\frac{J^m}{J-1} - \frac{J}{J-1}\frac{J^m - 1}{J-1}$$

On obtient donc :

$$\begin{aligned} n &\geq m(c - r + \frac{1}{J-1}) - \frac{J}{J-1}(c-r) + mr \\ &\geq (m-2)\left(c + \frac{1}{J-1}\right) - \frac{J}{J-1}c \\ &\geq (m-2)c \end{aligned}$$

De plus, on a $c < \frac{J^{m+1}-1}{J-1}$ donc $c < c(J-1) + 1 < J^{m+1}$ et $m + 1 \geq \log_J(c)$. Au final on obtient :

$$n > c \log_J \left(\frac{c}{J^3}\right). \tag{3.1}$$

Théorème 3.2.1 *Pour tout encodeur SPI à s états,*

$$\ell(y_1^n) \geq c(u_1^n) \log_2 \left(\frac{c(u_1^n)}{8s^2}\right).$$

Démonstration.

On a $u_1^1 = w_1 \dots w_c$ où $c = c(u_1^n)$ mots différents. On pose c_{ij} = le nombre de mots qui trouvent l'encodeur dans l'état i et le laissent dans l'état j. Comme l'encodeur est SPI, les sorties correspondantes sont nécessairement différentes et leur longueur totale ℓ_{ij} doit satisfaire (3.1) avec $J = 2$ puisque y est une suite binaire, donc :

$$\ell_{ij} \geq c_{ij} \log_2 \left(\frac{c_{ij}}{8}\right).$$

Donc $\ell(y_1^n) = \sum_{1 \leq i,j \leq s} c_{ij} \log_2 \left(\frac{c_{ij}}{8}\right)$. Comme $\sum c_{ij} = c(u_1^n)$ et que le minimum du terme de droite sous cette contrainte est atteint à $c_{ij} = \frac{c(u_1^n)}{s^2}$ (fonction convexe symétrique), on obtient le résultat voulu. □

Le lemme suivant est montré en exercie.

Lemme 3.2.1

$$c(u_1^n) = O(\frac{n}{\log(n)})$$

D'après le théorème,

$$\begin{aligned}
\rho_s(u) &\geq \limsup \frac{1}{n} c(u_1^n) \log_2(\frac{c(u_1^n))}{8s^2}) \\
&= \limsup_{n\to\infty} \frac{1}{n} c(u_1^n) \log_2(c(u_1^n)) - \limsup_{n\to\infty} \frac{1}{n} c(u_1^n) \log(8s^2) \\
&= \limsup_{n\to\infty} \frac{1}{n} c(u_1^n) \log_2(c(u_1^n)).
\end{aligned}$$

Comme cette dernière expression ne dépend pas de s, on a :

$$\rho(u) \geq \limsup_{n\to\infty} \frac{1}{n} c(u_1^n) \log_2(c(u_1^n)). \tag{3.2}$$

3.3 Algorithme de Lempel-Ziv

Nous décrivons maintenant l'algorithme de Lempel-Ziv. Le fonctionnement est le suivant :

- initialiser un dictionnaire avec tous les mots de longueur 1 ;
- attribuer à chaque mot un codage en binaire, par ordre lexicographique ; si le dictionnaire a D mots, la longueur des mot-code est $\lceil \log_2 D \rceil$.
- chaque fois qu'un mot de longueur m appartenant au dictionnaire est lu en entrée,
  - émettre le mot-code correspondant,
  - remplacer dans le dictionnaire le mot par l'ensemble des extensions d'une lettre du mot (c'est à dire les mots de longueur $m+1$ ayant comme préfixe le mot lu en entrée).

Un exemple permet de mieux saisir le comportement de l'algorithme. Soit *aaaccb* la chaîne à compresser. La figure 3.3 donne alors les différents états du dictionaire en lisant de gauche à droite. La chaîne émise est 00 000 110 0111.

Le décodage s'effectue de façon parfaitement symétrique. Étant donné que la mise à jour du dictionnaire est faite après l'émission du mot-code, le décodeur peut faire la même mise à jour une fois le mot-code reçu, et donc maintenir le même dictionnaire tout au long de la décompression.

On note qu'il demeure un problème lors de la fin du codage : le dernier mot lu ne correspond pas nécessairement à une feuille de l'arbre. Diverses solutions existent

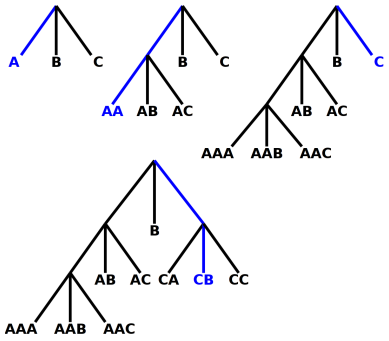

FIGURE 3.2 – Évolutions successives du dictionnaire

pour pallier à ceci, par exemple en choisisant de numéroter tous les nøeuds de l'arbre plutôt que les feuilles.

Voyons maintenant dans quelle mesure le codage de Lempel-Ziv est efficace. Si l'algorithme découpe u_1^n en $c_{LZ}(u_1^n)$ mots $w_1, w_2, \dots w_{c_{LZ}}$, alors $u_1^n = \varepsilon w_1, w_2, \dots w_{c_{LZ}}$ et les $c_{LZ}(u_1^n) - 1$ premiers mots sont différents. Si l'on concatène les deux derniers mots, on obtient un découpage en $c_{LZ}(u_1^n)$ mots différents. On a alors $c_{LZ}(u_1^n) \leq c(u_1^n)$.

La taille du dictionnaire à la fin du découpage de u_1^n est $J + (c_{LZ}(u_1^n) - 1)(J-1)$, et le nombre de nøeuds dans l'arbre $c_{lz}(u_1^n)J$.

Même si on attribue à chaque noeud de l'arbre un mot-code, le nombre total de digits binaires envoyés sera :

$$\begin{aligned} \ell_{LZ}(y_1^n) &\leq c_{LZ}(u_1^n)\lceil \log_2(Jc_{LZ}(u_1^n))\rceil \\ &\leq c(u_1^n)\log_2(2Jc(u_1^n) \end{aligned}$$

D'où

$$\limsup_{n\to\infty} \frac{1}{n}\ell_{LZ}(y_1^n) \leq \limsup_{n\to\infty} \frac{c(u_1^n)}{n}\log_2(c(u_1^n)) \leq \rho(u),$$

où la dernière inégalité vient de (3.2).

L'algorithme de Lempel-Ziv est donc au moins aussi bon que n'importe quel encodage SPI par automates finis. Noter cependant que l'algorithme de Lempel-Ziv n'est pas un encodage par automate fini. Par contre, le codage de Huffman par blocs fait partie de cette dernière classe et donc l'algorithme de Lempel-Ziv "fait donc aussi bien" que le codage de Huffman par blocs pour toute longueur de bloc.

3.4 Exercice : optimalité de l'algorithme de Lempel-Ziv pour une source sans mémoire

1. Montrer le lemme énoncé sans démonstration en cours : $c(u_1^n) = O(n/\log n)$.
   - En reprenant les notations du cours, soit $c' = c/J^3$ et $n' = n/J^3$. On a vu en cours que $n' > c' \log_J c'$. Pour n suffisament grand tel que $\sqrt{n'} \leq 2\frac{n'}{\log_J n'}$, on a
     - soit $c' < \sqrt{n'}$ et donc $c' < 2n'/\log_J n'$.
     - soit $c' \geq \sqrt{n'}$ et alors $c' < \frac{n'}{\log_J c'} \leq \frac{n'}{\log_J \sqrt{n'}} = \frac{2n'}{\log_J n'}$.
2. Montrer que pour toute v.a. X à valeurs entières de moyenne $E[X]$, on a :
   $$H(X) \leq (E[X]+1)\log(E[X]+1) - E[X]\log E[X],$$
   avec égalité quand X suit une loi géométrique : $P(X=k) = q^k(1-q)$ pour $k \geq 0$.
   - pour Y de loi géométrique $P(Y=k) = q^k(1-q)$ pour $k \geq 0$, on a $E[Y] = \frac{q}{1-q}$ et :
     $$\begin{aligned} H(Y) &= -\sum_k q^k(1-q)\log q^k(1-q) \\ &= -\log(1-q) - E[Y]\log q, \end{aligned}$$
     on a donc bien égalité dans ce cas.
   - Soit X une v.a. discrète de loi p_k et Y de loi géométrique de même moyenne q_k. On a alors
     $$\begin{aligned} H(X) &= -\sum p_k \log p_k \\ &= -\sum p_k \log \frac{p_k}{q_k} - \sum p_k \log q_k \\ &= D(p||q) - \sum p_k \log q^k(1-q) \\ &\leq -\log(1-q) - \sum k p_k \log q = H(Y), \end{aligned}$$
     où l'inégalité vient de $D(p||q) \geq 0$.
3. Soit $\{U_i\}_{i=1}^{\infty}$ une suite de v.a. i.i.d. à valeurs dans un ensemble fini $\mathcal{U}$ et d'entropie $H(U)$. On note $C(n)$ la v.a. égale au nombre maximal de mots distincts en lequel $U^{(n)}$ peut être découpé. C'est à dire avec les notations du cours : $C(n) = c(U_1^n)$. Nous allons montrer qu'avec probabilité 1, on a :
   $$\limsup_{n\to\infty} \frac{C(n)\log C(n)}{n} \leq H(U).$$
   Ce résultat permet de démontrer l'optimalité de l'algorithme de Lempel-Ziv dans le cas particulier d'une source sans mémoire.

a) Pour un découpage de u_1^n en c mots distincts, on note c_ℓ le nombre de mots de longueur ℓ. Montrer que

$$\log P(u_1, u_2, \ldots, u_n) \leq -\sum_\ell c_\ell \log c_\ell.$$

b) On définit la v.a. Z par $P(Z = \ell) = \frac{c_\ell}{c}$. En utilisant les deux exercices précédents, montrer que

$$\lim_{n\to\infty} \frac{c}{n} H(Z) = 0$$

c) Conclure.

a) On note le découpage : $u_1 u_2 \ldots u_n = w_1 w_2 \ldots w_c$. On a alors :

$$\begin{aligned}
\log P(u_1, u_2, \ldots, u_n) &= \sum_{i=1}^{c} \log P(w_i) \\
&= \sum_\ell \sum_{i,|w_i|=\ell} \log P(w_i) \\
&\leq \sum c_\ell \log \left(\sum_{i,|w_i|=\ell} \frac{P(w_i)}{c_\ell} \right) \\
&\leq -\sum_\ell c_\ell \log c_\ell,
\end{aligned}$$

où la première inégalité provient de la concavité du log et la seconde du fait que le w_i étants distincs $\sum_i P(w_i) \leq 1$.

b) Z est une v.a. de moyenne $E[Z] = \frac{\sum_\ell \ell c_\ell}{c} = \frac{n}{c}$. On a donc :

$$\begin{aligned}
H(Z) &\leq \left(\frac{n}{c} + 1\right) \log \left(\frac{n}{c} + 1\right) - \frac{n}{c} \log \frac{n}{c} \\
&= \log \left(\frac{n}{c} + 1\right) + \frac{n}{c} \log \left(\frac{c}{n} + 1\right).
\end{aligned}$$

Donc

$$\frac{c}{n} H(Z) \leq \frac{c}{n} \log \left(\frac{n}{c} + 1\right) + \log \left(\frac{c}{n} + 1\right),$$

le résultat découle de $c = O(n/\log n)$.

c) Pour tout $u_1, \ldots, u_n$, et tout découpage en c mots distincst, on a

$$\begin{aligned}
\log P(u_1, \ldots, u_n) &\leq -\sum_\ell c_\ell \log c_\ell \\
&= -c \log c - c \sum_\ell \frac{c_\ell}{c} \log \frac{c_\ell}{c} \\
&= -c \log c - c H(Z).
\end{aligned}$$

On a donc avec probabilité 1 :

$$-\frac{1}{n}\log P(U_1,\ldots,U_n) \;\geq\; \frac{C(n)}{n}\log C(n) - \frac{C(n)}{n}H(Z),$$

et on obtient le résultat désiré en prenant la limite $n \to \infty$.

Chapitre 4

Propriétés de l'entropie et de l'information mutuelle

Notations On utilise les conventions : $0 \times \log 0 = 0$, $a \times \log \frac{a}{0} = \infty$ pour tout $a > 0$ et $0 \times \log \frac{0}{0} = 0$.

4.1 Rappels

$(X, Y) \sim p(x, y) = \mathbb{P}(X = x, Y = y)$

Définition 4.1.1 *entropie (jointe) :* $H(X, Y) = -\sum_{x \in X} \sum_{y \in Y} p(x, y) \log p(x, y)$ *(logarithme en base 2)*

entropie conditionnelle :

$$\begin{aligned} H(Y|X) &= \sum_{x \in X} p(x) H(Y|X = x) \\ &= -\sum_{x \in X} p(x) \sum_{y \in Y} p(y|x) \log p(y|x) \\ &= -\sum_{x \in X} \sum_{y \in Y} p(x, y) \log p(y|x). \end{aligned}$$

Théorème 4.1.1 ***règle de la chaîne***

$$H(X, Y) = H(X) + H(Y|X)$$

Démonstration. On utilise $p(x,y) = p(x)p(y|x)$ de telle sorte que :

$$\begin{aligned} H(X,Y) &= -\sum_{x\in X}\sum_{y\in Y} p(x,y)\log p(x,y) \\ &= -\sum_{x\in X}\sum_{y\in Y} p(x,y)\log p(x) - \sum_{x\in X}\sum_{y\in Y} p(x,y)\log p(y|x) \\ &= H(X) + H(Y|X). \end{aligned}$$

□

Corollaire 4.1.1

$$H(X,Y|Z) = H(X|Z) + H(Y|X,Z)$$

Définition 4.1.2 *L'information mutuelle est définie par :*

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) \\ &= H(Y) - H(Y|X) \\ &= D(p(x,y)||p(x)p(y)), \end{aligned}$$

avec $D(p||q) = \sum_{x\in X} p(x) log \frac{p(x)}{q(x)}$ *la distance de Kullbak-Leibler.*

4.2 Règles de la chaîne

Théorème 4.2.1 ***(pour l'entropie)*** *Soit* $(X_1, ..., X_n) \sim p(x_1, ..., x_n)$*, alors*

$$H(X_1, ..., X_n) = \sum_{i=1}^{n} H(X_i|X_{i-1}...X_1)$$

Démonstration. On écrit $p(x_1, \ldots, x_n) = p(x_1)p(x_2|x_1)...p(x_n|x_{n-1}...x_1)$, soit :

$$\begin{aligned} H(X_1, ..., X_n) &= -\sum_{x_1...x_n} p(x_1, ..., x_n) log \prod_{i=1}^{n} p(x_i|x_{i-1}...x_1) \\ &= -\sum_{i=1}^{n}\sum_{x_1...x_n} p(x_1, ..., x_n) log p(x_i|x_{i-1}...x_1) \\ &= \sum_{i=1}^{n} H(X_i|X_{i-1}...X_1) \end{aligned}$$

□

Définition 4.2.1 *l'information mutuelle conditionnelle des v.a X et Y étant donné Z est : $I(X;Y|Z) = H(X|Z) - H(X|Y,Z)$.*

Théorème 4.2.2 ***(pour l'information)***

$$I(X_1, ..., X_n; Y) = \sum_{i=1}^{n} I(X_i; Y|X_{i-1}..X_1)$$

Démonstration.

$$\begin{aligned} I(X_1...X_n;Y) &= H(X_1, ..., X_n) - H(X_1, ..., X_n|Y) \\ &= \sum_{i=1}^{n} H(X_i|X_{i-1}..X_1) - \sum_{i=1}^{n} H(X_i|X_{i-1}...X_1, Y) \\ &= \sum_{i=1}^{n} I(X_i; Y|X_{i-1}..X_1). \end{aligned}$$

□

4.3 Inégalités de convexité

Théorème 4.3.1 *$D(p||q) \geq 0$ avec égalité ssi $\forall x$, $p(x) = q(x)$.*

Démonstration. Par stricte concavité du log, on a :

$$\sum_{x,\, p(x)>0} p(x) \log \frac{q(x)}{p(x)} \leq \log \sum_{x,\, p(x)>0} q(x) \leq 0.$$

□

Corollaire 4.3.1 *$I(X;Y) \geq 0$ avec égalité ssi X et Y sont indépendantes*

Démonstration. Il suffit d'observer que : $I(X;Y) = D(p(x,y)||p(x)p(y))$ □

Corollaire 4.3.2
- *$D(p(y|x)||q(y|x)) \geq 0$ avec égalité ssi $p(y|x) = q(y|x), \forall y, x$ tq $p(x) > 0$.*
- *$I(X;Y|Z) \geq 0$ avec égalité ssi X et Y sont indépendantes conditionnellement à Z.*

Théorème 4.3.2 *$H(X) \leq log|\mathcal{X}|$ où $|\mathcal{X}|$ = nombre d'éléments dans le support de X, c'est à dire le nombre de x tels que $p(x) > 0$ avec égalité ssi $X \sim Unif(\mathcal{X})$.*

Démonstration. Soit $u(x) = \dfrac{1}{|\mathcal{X}|}$ la distribution uniforme sur $\mathcal{X}$. On note alors que

$$D(p||u) = \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{u(x)} = log|\mathcal{X}| - H(X)$$

□

Théorème 4.3.3 *$H(X|Y) \leq H(X)$ avec égalité ssi X et Y sont indépendantes.*

Démonstration. $0 \leq I(X;Y) = H(X) - H(X|Y)$. □

EXEMPLE 4.3.1:

Attention l'entropie conditionnelle diminue en moyenne mais on peut avoir $H(X|Y = Y) > H(X)$ pour des y particuliers, comme le montre l'exemple suivant :

$X \setminus Y$	1	2
1	0	$\frac{3}{4}$
2	$\frac{1}{8}$	$\frac{1}{8}$

On a $H(X) = H(1/8) \approx 0.544$ bit. $H(X|Y = 1) = 0$ bit et $H(X|Y = 2) = 1$ bit. L'incertitude sur X augmente si $Y = 2$ est observée et elle diminue (fortement !) si $Y = 1$ est observée. En moyenne, on a :

$$H(X|Y) = 3/4H(X|Y = 1) + 1/4H(X|Y = 2) = 1/4 \leq H(X).$$

Théorème 4.3.4 *$H(X_1, ..., X_n) \leq \sum_{i=1}^{n} H(X_i)$ avec égalité ssi les X_i sont indépendantes.*

Démonstration.

$$H(X_1, ...X_n) = \sum_{i=1}^{n} H(X_i|X_{i-1}...X_1) \leq \sum_{i=1}^{n} H(X_i)$$

avec égalité ssi X_i est indépendante de $X_{i-1}, \ldots X_1$, c'est à dire si les X_i sont indépendantes entre elles. □

4.3.1 Inégalité logsum et applications

Théorème 4.3.5 *pour tout, $a_1...a_n b_1...b_n \geq 0$, on a :*

$$\sum_{i=1}^{n} a_i \log \frac{a_i}{b_i} \geq \sum_{i=1}^{n} a_i \log \frac{\sum_{i=1}^{n} a_i}{\sum_{i=1}^{n} b_i}$$

avec égalité ssi $\frac{a_i}{b_i} = cst.$

Démonstration. On suppose tout d'abord que pour tout i, $a_i > 0$ et $b_i > 0$. Soit $f(t) = t \log t$. On a $f'(t) = \frac{\log t}{+} \frac{1}{\log(e)}$ et $f''(t) = \frac{1}{t \log e} > 0$ pour tout $t > 0$. Donc f est strictement convexe et $\sum \alpha_i f(t_i) \leq f(\sum \alpha_i t_i)$, pour $\alpha_i \geq 0$, $\sum \alpha_i = 1$, $t_i > 0$. Il suffit alors de prendre

$$\alpha_i = \frac{b_i}{\sum b_j} \text{ et, } t_i = \frac{a_i}{b_i}.$$

Si $a_j = 0$ et $b_j = 0$ alors avec les conventions choisies, le j-ème terme du membre de gauche est nul et on peut supprimer a_j et b_j à gauche et à droite.

Si $a_j > 0$ et $b_j = 0$, le terme de gauche vaut ∞ donc l'inégalité est toujours valable.

Si $a_j = 0$ et $b_j > 0$, on peut supprimer le j-ème terme dans le mebre de gauche et en supprimant b_j dans le terme de droite, on obtient bien un majorant :

$$\sum_{i \neq j} a_i \log \frac{a_i}{b_i} \geq \sum_{i \neq j} a_i \log \frac{\sum_{i \neq j} a_i}{\sum_{i \neq j} b_i} \geq \sum_{i=1}^{n} a_i \log \frac{\sum_{i=1}^{n} a_i}{\sum_{i=1}^{n} b_i}.$$

□

Lemme 4.3.1 *$D(p||q)$ est convexe en la paires (p, q), c'est à dire, pour tout $\lambda \in \{0, 1\}$*

$$D(\lambda p_1 + (1-\lambda)p_2 || \lambda q_1 + (1-\lambda)q_2) \leq \lambda D(p_1||q_1) + (1-\lambda) D(p_2||q_2)$$

Démonstration. On applique l'inégalité logsum à x fixé :

$$(\lambda p_1(x) + (1-\lambda)p_2(x)) \log \frac{\lambda p_1(x) + (1-\lambda)p_2(x)}{\lambda q_1(x) + (1-\lambda)q_2(x)} \leq \lambda p_1(x) \log \frac{p_1(x)}{q_1(x)} + (1-\lambda) p_2(x) \log \frac{p_2(x)}{q_2(x)}.$$

□

Théorème 4.3.6 *$H(X)$ est une fonction concave de $p(x)$.*

Démonstration. Il suffit de noter que : $H(X) = log|\mathcal{X}| - D(p||u)$, où u les distribution uniforme. □

Théorème 4.3.7 *Soit $(X,Y) \sim p(x,y)$. L'information mutuelle $I(X,Y)$ est une fonction concave de $p(x)$, à $p(y|x)$ fixée, et convexe en $p(y|x)$, à $p(x)$ fixée.*

Démonstration. $I(X;Y) = H(Y) - \sum_x p(x)H(Y|X=x)$

Si $p(y|x)$ est fixée alors $p(y)$ est une fonction linéaire de $p(x)$. Donc $H(Y)$ qui est concave en $p(y)$ est concave en $p(x)$. Le second terme est linéaire en $p(x)$ donc la différence est concave.

Maintenant $p(x)$ est fixée. A $p_1(y|x)$ et $p_2(y|x)$, on associe :

$$\begin{aligned} p_1(x,y) &= p_1(y|x)p(x) \text{ et, } p_1(y) \\ p_2(x,y) &= p_2(y|x)p(x) \text{ et, } p_2(y). \end{aligned}$$

On définit alors $p_\lambda(y|x) = \lambda p_1(y|x) + (1-\lambda)p_2(y|x)$ ainsi que $p_\lambda(x,y) = p(x)p_\lambda(y|x) = \lambda p_1(x,y) + (1-\lambda)p_2(x,y)$ et $p_\lambda(y)$. Finalement, soit $q_\lambda(x,y) = p(x)p_\lambda(y)$ de telle sorte que $q_\lambda(x,y) = \lambda q_1(x,y) + (1-\lambda)q_2(x,y)$ et

$$I(X;Y) = D(p_\lambda(x,y)||q_\lambda(x,y)),$$

comme $D(p||q)$ est convexe en (p,q), $I(X,Y)$ est convexe en $p(y|x)$ à $p(x)$ fixée. □

4.4 Data Processing inequality

Définition 4.4.1 *(X,Y,Z) est une chaîne de Markov si $p(z|x,y) = p(z|y)$. On a donc $p(x,y,z) = p(x)p(y|x)p(z|y)$. On notera parfois $X \to Y \to Z$.*

Remarque 4.4.1 *$p(x,z|y) = \dfrac{p(x,y,z)}{p(y)} = \dfrac{p(x,y)}{p(y)} \times p(z|y) = p(x|y)p(z|y)$ donc $X \to Y \to Z$ ssi X et Z sont conditionnellement indépendantes étant donné Y donc $Z \to Y \to X$.*

Théorème 4.4.1 *Si $X \to Y \to Z$ alors* $\max\{I(X;Y), I(X;Z)\} \geq I(X;Z)$.

Démonstration.

$$\begin{aligned} I(X;Y,Z) &= I(X;Z) + I(X;Y|Z) \\ &= I(X;Y) + I(X;Z|Y). \end{aligned}$$

Comme X et Z sont conditonnellement indépendantes, on a $I(X;Z|Y) = 0$. Comme $I(X;Y|Z) \geq 0$, on a $I(X;Y) \geq I(X;Z)$. L'autre inégalité provient du fait que $Z \to Y \to X$ et de la symétrie de l'information mutuelle. □

Corollaire 4.4.1 *Si $X \to Y \to Z$ alors $I(X;Y|Z) \leq I(X;Y)$.*

Démonstration. En reprenant la décomposition de $I(X;Y,Z)$ ci-dessus, il suffit de noter que $I(X;Z) \geq 0$. □

4.5 Inégalité de Fano

On veut estimer $X \sim p(x)$ (p(x)>0 pour tout $x \in \mathcal{X}$) en observant Y avec $p(y|x)$. Soit $\hat{X}$ un estimateur de X, c'est à dire tel que $X \to Y \to \hat{X}$.

Théorème 4.5.1 *Pour tout $\hat{X}$ tel que $X \to Y \to \hat{X}$, soit $P_e = P(X \neq \hat{X})$ alors on a*

$$H(P_e) + P_e log(|\mathcal{X}| - 1) \geq H(X|\hat{X}) \geq H(X|Y),$$

où $H(p) = -p \log p - (1-p)\log(1-p)$.

Remarque 4.5.1 *$P_e = 0 \to H(X|Y) = 0 \to Y$ est une fonction de X.*

Démonstration. On définit :

$$E = \begin{cases} 1 & \text{si } \hat{X} \neq X \\ 0 & \text{sinon} \end{cases}$$

On a :

$$\begin{aligned} H(E, X|\hat{X}) &= H(X|\hat{X}) + H(E|X, \hat{X}) \\ &= H(E|\hat{X}) + H(X|E, \hat{X}). \end{aligned}$$

Comme $H(E|X,\hat{X}) = 0$ et $H(E|\hat{X}) \leq H(E) = H(P_e)$, et de plus,

$$\begin{aligned} H(X|E,\hat{X}) &= P(E=0)H(X|\hat{X}, E=0) + P(E=1)H(X|\hat{X}, E=1) \\ &\leq (1-P_e) \times 0 + P_e \log(|\mathcal{X}| - 1), \end{aligned}$$

au final, on obtient :

$$H(P_e) + P_e \log(|\mathcal{X}| - 1) \geq H(X|\hat{X}).$$

Par l'inégalité 'data processing', on a : $I(X;\hat{X}) \leq I(X;Y)$ et donc $H(X|\hat{X}) \geq H(X|Y)$. □

4.6 Exercice : Lien entre théorie de l'information et courses de chevaux

Objectif : définir une stratégie 'optimale' pour jouer aux courses.

Problème : M chevaux concurrents, p_i proba le $i$ème cheval gagne. Si le $i$ème cheval gagne, on gagne $o(i)$ fois sa mise. C'est à dire, si on a misé 1 euro, on obtient $o(i)$ euro si i gagne et rien sinon (la mise de 1 euro est perdue).

On suppose que le joueur distribue toute sa fortune sur les chevaux. Soit $b(i) =$ la fraction parié sur le cheval i. On a $b(i) \geq 0$ et $\sum_{i=1}^{M} b(i) = 1$.

Après n courses, la fortune du joueur est (en supposant qu'initialement $S_0 = 1$), $S_n = \prod_{i=1}^{n} S(X_i)$ avec $S(X) = b(X)o(X)$ et X est le cheval vainqueur.

Théorème 4.6.1 *Si les X_i sont i.i.d. $\sim p(x)$ alors la fortune du joueur utilisant la stratégie b "croît" exponentiellement : $lim \frac{1}{n} \log_2 S_n = W(b,p) = \sum_{k=1}^{m} p_k \log_2(b(k)o(k))$ où la convergence est en probabilité.*

Démonstration. Il suffit d'appliquer la loi faible des grands nombres et de noter que $W(b,p) = E[\log S(X)]$. □

Attention rien ne dit que $W(b,p) \geq 0$!

Théorème 4.6.2 ***(Critère de Kelly)*** *La stratégie optimale est de prendre $b^* = p$ et dans ce cas $W(b^*,p) = \sum p_i \log o(i) - H(p)$ (optimum doubling rate)*

Démonstration. Il suffit d'écire :

$$W(b,p) \quad = \quad \sum p_i \log o(i) - H(p) - D(p||b).$$

□

Dans le cas équitable : $\sum_i \frac{1}{o(i)} = 1$ alors $r_i = \frac{1}{o(i)}$ est une distribution de probabilité qui correspond à l'estimée de la probabilité p_i par le bookmaker. On a alors

$$W(b,p) = D(p||r) - D(p||b)$$

Si notre estimée b de p est meilleure que celle du bookmaker alors $W(b,p) \geq 0$.

Dans le cas spécial où $o_i = m$ pour tout i, on a $W(b^*,p) = \log m - H(p)$. Les courses à faible entropie sont les plus profitables.

On considère maintenant le cas où le joueur peut choisir de ne pas miser une fraction de sa fortune b_0.

a) Si les cotes sont équitables, c'est à dire $\sum_{i=1}^{m} \frac{1}{o_i} = 1$, la stratégie optimale consiste à tout parier de façon proportionnelle a p.

b) Si les cotes sont favorables, $\sum_{i=1}^{m} \frac{1}{o_i} < 1$, montrer qu'il est possible de parier sans risque, c'est à dire la fortune du joueur augmente à chaque course avec probabilité 1.

c) Dans le cas favorable, quelles est la stratégie optimale ?

a) Les stratégies $b_0 > 0$, $b_i =$ et $b'_0 = 0$, $b'_i = b_i + \frac{b_0}{o_i}$ sont équivalentes. Pour l'étude des stratégies optimales, on peut donc se restreindre aux stratégies avec $b_0 = 0$ qui ont été étudiées en cours.

b) Soit $c = \left(\sum_{i=1}^{m} \frac{1}{o_i}\right)^{-1} > 1$. La stratégie $b_i = \frac{c}{o_i}$ permet d'avoir sa fortune doublée de $\log_2 c > 0$ quelque soit l'issue de la course.

c) Nous allons montrer que toute stratégie avec $b_0 > 0$ est sous-optimale. Le résultat du cours permet alors de conclure que la stratégie optimale est de parier proportionellement à p. Le 'doubling rate' d'une stratégie b est donné par

$$\sum_{k=1}^{m} p_k \log_2 (b_0 + b_k o_k) .$$

Si $b_0 > 0$, on considère alors la stratégie $b'_0 = 0$ et $b'_k = b_k + b_0 \frac{c}{o_k}$ qui a un 'doubling rate' de

$$\sum_{k=1}^{m} p_k \log_2 (b_0 c + b_k o_k) ,$$

qui est plus grand que le précédent puisque $c > 1$.

Chapitre 5

Canaux discrets sans mémoire et leurs fonctions capacité-coût

5.1 Définitions et codes sans erreur

Un canal discret sans mémoire est caractérisé par :
- un alphabet d'entrée $\mathcal{X}$, de cardinal r
- un alphabet de sortie $\mathcal{Y}$, de cardinal s
- un coût $b(x)$ associé à chaque entrée x
- une probabilité de transition $p(y|x)$ qui définit une matrice stochastique (notée Q) de taille $r \times s$:

$$p(y|x) \geq 0, \quad \forall x \in \mathcal{X}, \sum_{y \in \mathcal{Y}} p(y|x) = 1.$$

Ce modèle de canal reçoit donc une entrée x à chaque unité de temps et émet y avec probabilité $p(y|x)$ pour un coût de $b(x)$.

Canal sans mémoire : s'il est utilisé n fois sur les entrées $x_1, \dots x_n$, la probabilité qu'il émette $y_1 \dots y_n$ est

$$p(\underline{y}|\underline{x}) = \prod_{i=1}^{n} p(y_i|x_i)$$

et cela coute

$$b(\underline{x}) = \sum_{i=1}^{n} b(x_i).$$

Si les entrées sont des variables aléatoires (v.a.) $\underline{X} = (X_1, \ldots, X_n)$ suivant une loi $p(\underline{X})$, alors le coût moyen est

$$E\left[b(\underline{X})\right] = \sum_{\underline{x}} p(\underline{x}) b(\underline{x}).$$

Définition 5.1.1 *Un (M, n)-code est un sous-ensemble*

$$C = \{\underline{x}_1, \ldots \underline{x}_M\} \subset \mathcal{X}^n.$$

La longueur d'un tel code est n.
Le taux du code est $R = \frac{\log_2 M}{n}$ (bits/symbole).
Un code est dit β-admissible si $\forall i, b(\underline{x}_i) \leq n\beta$.
Une règle de décodage pour ce code est une fonction

$$f : \mathcal{Y}^n \longrightarrow C \cup \{?\}$$

La *probabilité d'erreur* sur le mot code $\underline{x}_i$ est

$$\begin{aligned} P_E^{(i)} &= P\left(f(\underline{y}) \neq \underline{x}_i | \underline{x}_i \text{ envoyé}\right) \\ &= \sum_{f(\underline{y}) \neq \underline{x}_i} p(\underline{y}|\underline{x}_i) \end{aligned}$$

EXEMPLE 5.1.1: Prenons $\mathcal{X} = \{0, 1/2, 1\}$, $\mathcal{Y} = \{0, 1\}$ et $b(0) = b(1) = 1, b\left(1/2\right) = 0$ et les probabilités de transition suivantes :

$$Q = \begin{pmatrix} 1 & 0 \\ 1/2 & 1/2 \\ 0 & 1 \end{pmatrix}.$$

Soit $k \leq n$. L'ensemble suivant est un $(2^k, n)$-code

$$C = \left\{(x_1 \cdots x_k, 1/2 \cdots 1/2) \,| x_i \in \{0, 1\}\right\}.$$

Ce code est β-admissible pour $\beta \geq \frac{k}{n}$ et son taux est $\frac{k}{n}$ bits/symbole.
En prenant la règle de décodage :

$$f(y_1, \ldots, y_n) = \left(y_1, \ldots, y_k, 1/2 \ldots 1/2\right),$$

la probabilité d'erreur pour ce canal est nulle (pour $k \leq n$) quelque soit le mot code envoyé.

Réciproquie du Théorème de codage de canal pour une probabilité d'erreur nulle

Supposons qu'on ait un $(2^{nR}, n)$-code avec $\forall i, P_E^{(i)} = 0$, et $\underline{X}$ est uniformément distribué. On a alors :

$$\begin{aligned}
nR = logM &= H(\underline{X}) \\
&= H(\underline{X}|\underline{Y}) + I(\underline{X};\underline{Y}) \\
&= 0 + \left(H(\underline{Y}) - \sum_{i=1}^{n} H\left(Y_i|Y_1,\ldots,Y_{i-1},\underline{X}\right) \right) \qquad \text{car } \forall i, P_E^{(i)} = 0, \\
&= H(\underline{Y}) - \sum_{i=1}^{n} H\left(Y_i|X_i\right) \qquad \text{car canal sans mémoire,} \\
&\leq \sum_{i=1}^{n} H\left(Y_i\right) - \sum_{i=1}^{n} H\left(Y_i|X_i\right) \\
&= \sum_{i=1}^{n} I\left(X_i;Y_i\right) \\
&\leq n \cdot max_{p(\underline{X})}\{I(\underline{X};\underline{Y})\} = nC
\end{aligned}$$

Cette dernière quantité s'appelle la capacité du canal.

5.2 Inégalité de Fano et réciproque du théorème de codage de canal

Le calcul précédent a été fait pour une probabilité d'erreur nulle. Il est facile de le généraliser comme suit.

On Suppose toujours que $\underline{X}$ uniformément distribuée. Donc l'erreur moyenne est définie par : $\overline{P_E} = \frac{1}{2^{nR}} \sum_i P_E^{(i)}$. On a alors,

$$\begin{aligned}
nR &= H(\underline{X}) \\
&= H(\underline{X}|\underline{Y}) + I(\underline{X},\underline{Y}) \\
&\leq 1 + P\left(f(\underline{Y}) \neq \underline{X}\right) nR + I(\underline{X};\underline{Y}) \qquad \text{par l'inégalité de Fano} \\
&\leq 1 + nR\overline{P_E} + I(\underline{X},\underline{Y}) \\
&\leq 1 + nR\overline{P_E} + nC
\end{aligned}$$

Ainsi, $\overline{P_E} \geq 1 - \frac{C}{R} - \frac{1}{nR}$. Par conséquent, si $R > C$, alors $\overline{P_E} > 0$ pour n suffisament grand et donc pour tout n (puisque dans le cas contraire, il suffirait alors de concatener un code court sans erreur pour obtenir une contradiction).

5.3 La fonction capacité-coût

Pour tout n, on définit la $n^{\text{ème}}$ fonction capacité coût par

$$C_n(\beta) = \max \left\{ I(\underline{X}, \underline{Y}) \,\middle|\, E\left[b(\underline{X})\right] \leq \beta \text{ et } P(\underline{Y}|\underline{X}) = \prod p(y_i|x_i) \right\}$$

Dans ce contexte, on dira que $\underline{X}$ est une source test et qu'elle est β-admissible si $E[b(\underline{X})] \leq n\beta$.

Remarque 5.3.1 *a) Pour $p(y|x)$ fixées, $I(X,;Y)$ est une fonction continue de $p(x)$ et la maximisation est faite sur un compacte donc atteinte.*
b) $C_n(\beta)$ est définit pour tout $\beta \geq \beta_{\min} = \min\{b(x)|x \in \mathcal{X}\}$.
c) C_n est croissante pour $\beta \geq \beta_{\min}$.

La fonction capacité-coût du canal est définie par

$$C_\beta = \sup_n \frac{1}{n} C_n(\beta)$$

Théorème 5.3.1 *C_n est concave en $\beta \geq \beta_{\min}$.*

Démonstration.
Soit $\alpha_1, \alpha_2 \geq 0$ tq $\alpha_1 + \alpha_2 = 1$.
On doit montrer que pour $\beta_1, \beta_2 \geq \beta_{\min}$, on a :

$$C_n(\alpha_1\beta_1 + \alpha_2\beta_2) \geq \alpha_1 C_n(\beta_1) + \alpha_2 C_n(\beta_2).$$

Soit $\underline{X}_1$ et $\underline{X}_2$ deux sources test de distribution p_1 et p_2 qui atteignent $C_n(\beta_1)$ et $C_n(\beta_2)$, c'est à dire telles que $\underline{Y}_1$ et $\underline{Y}_2$ soient les sorties associées et :
– $\underline{X}_i$ est β_i-admissible pour $i = 1, 2$;
– $I(\underline{X}_i; \underline{Y}_i) = C_n(\beta_i)$ pour $i = 1, 2$.
La source test $\underline{X}$ définie par la distribution $\alpha_1 p_1 + \alpha_2 p_2$ est $(\alpha_1\beta_1 + \alpha_2\beta_2)$-admissible, et donc $C_n(\alpha_1\beta_1 + \alpha_2\beta_2) \geq I(\underline{X}, \underline{Y})$, si on note $\underline{Y}$ la sortie associée. Comme $I(\underline{X}, \underline{Y})$ est concave en $p(x)$, on obtient :

$$\begin{aligned} I(\underline{X}, \underline{Y}) &\geq \alpha_1 I(\underline{X}_1, \underline{Y}_1) + \alpha_2 I(\underline{X}_2, \underline{Y}_2) \\ &= \alpha_1 C_n(\beta_1) + \alpha_2 C_n(\beta_2). \end{aligned}$$

□

Théorème 5.3.2 *Pour un canal discret sans mémoire,*

$$\forall n \geq 1, \forall \beta \geq \beta_{\min}, C_n(\beta) = n \cdot C_1(\beta)$$

Démonstration. Soit (X, Y) un couple de source test - sortie atteignant $C_1(\beta)$. En considérant la source test $\underline{X} = (X_1, \ldots, X_n)$ où $X, X_1, \ldots X_n$ sont i.i.d., on obtient facilement $C_n(\beta) \geq nC_1(\beta)$.

Pour ce qui est de l'autre sens de l'inégalité, soit $\underline{X}$ β-admissible qui atteint $C_n(\beta)$. Le canal n'ayant pas de mémoire, on a grâce au calcul fait en Section 5.1 :

$$C_n(\beta) = I(\underline{X}, \underline{Y}) \leq \sum_{i=1}^{n} I\left(X_i, Y_i\right).$$

Si on pose $\beta_i = Eb(X_i$, alors $\sum_i \beta_i \leq n\beta$.

Par définition, $\forall i, I(X_i, Y_i) \leq C_1(\beta_i)$, et on a donc :

$$\begin{aligned} C_n(\beta) &\leq n \cdot \frac{1}{n} \sum_{i=1}^{n} C_i(\beta_i) \\ &\leq n \cdot C_1\left(\frac{1}{n} \sum_{i=1}^{n} \beta_i\right) \qquad \text{(concavité de } C_1\text{)} \\ &\leq n \cdot C_1(\beta) \qquad \text{(croissance de } C_1\text{)} \end{aligned}$$

□

Corollaire 5.3.1 *Pour un canal discret sans mémoire (DMC en anglais discrete memoryless channel), $C(\beta) = C_1(\beta)$.*

Propriétés générales de $C(\beta)$ pour un DMC :

- $C(\cdot)$ est croissante et concave pour $\beta \geq \beta_{\min}$. Donc $C(\cdot)$ est continue pour $\beta > \beta_{min}$ (en exercice, on montre $C(\beta)$ est continue en $\beta_{\min}$).
- La capacité du canal est le maximum de la fonction capacité-coût du canal (cela revient à accepter un coût infini). Si on note celle-ci $\mathcal{C}_{\max}$ et $\beta_{\max} = \min\{E[b(\underline{X})] | I(\underline{X}, \underline{Y}) = C_{max}\}$, alors on voit que $C(\beta) = C_{\max}$ pour $\beta \geq \beta_{\max}$ et $C(\beta) < C_{\max}$ pour $\beta < \beta_{\max}$. Donc $\beta \mapsto C(\beta)$ est strictement croissante sur $(\beta_{\min}, \beta_{\max})$. En particulier, on a pour $\beta \in [\beta_{\min}, \beta_{\max}]$:

$$C(\beta) = \max\{I(X;Y) | E[b(X)] = \beta\}$$

EXEMPLE 5.3.1: CANAL BINAIRE SYMMÉTRIQUE

Prenons $\mathcal{X} = \mathcal{Y} = \{0, 1\}$, $b(0) = 0$ et $b(1) = 1$ et les probabilités de transition suivantes :

$$Q = \begin{pmatrix} q & p \\ p & q \end{pmatrix},$$

avec $p, q \geq 0$ tels que $p + q = 1$ et $p \leq 1/2$.
Alors $\beta_{\min} = 0$ et $C_{\min} = 0$.

Soit X une source test qui atteint $C(\beta)$ pour $0 \leq \beta \leq \beta_{\max}$, cette dernière quantité étant pour le moment inconnue. On a en particulier

$$\begin{aligned} P(X = 1) &= b(1)P(X = 1) + b(0)P(X = 0) \\ &= Eb(X) \\ &= \beta \end{aligned}$$

et

$$\begin{aligned} C(\beta) &= I(X, Y) \\ &= H(Y) - H(Y|X) \\ &= H\left(Ber_{(1-\beta)p+\beta q}\right) - H\left(Ber_p\right) \\ &\leq 1 - H\left(Ber_p\right) \end{aligned}$$

avec égalité pour $\beta_{\max} = 1/2$, donc $C_{\max} = 1 - H\left(Ber_p\right) = 1 - H(p)$. Au final, on a :

$$C(\beta) = \begin{cases} H((1-\beta)q + \beta p) - H(p) & 0 \leq \beta \leq 1/2 \\ 1 - H(p) & \beta > 1/2. \end{cases}$$

EXEMPLE 5.3.2: Reprenons l'exemple avec $\mathcal{X} = \{0, 1/2, 1\}$, $\mathcal{Y} = \{0, 1\}$ et $b(0) = b(1) = 1$, $b\left(1/2\right) = 0$ et les probabilités de transition suivantes :

$$Q = \begin{pmatrix} 1 & 0 \\ 1/2 & 1/2 \\ 0 & 1 \end{pmatrix}.$$

Alors $\beta_{min} = 0$ et $C_{min} = 0$.

Soit X une source test qui atteint $C(\beta)$ pour $0 \leq \beta \leq \beta_{\max}$. On a $\beta = Eb(X) = p(0) + p(1)$ et $I(X; Y)$ est concave en $p(0), p(1/2), p(1)$, donc par symmétrie $p(0) = p(1) = \beta_2$, on a alors :

$$C(\beta) = I(X, Y) = H(Y) - H(Y|X) = 1 - (1 - \beta) = \beta.$$

Donc $\beta_{\max} = 1$ et

$$\begin{aligned} \forall \beta \in [0,1] \quad & C(\beta) = \beta \\ \forall \beta \in [1,\infty[\quad & C(\beta) = 1. \end{aligned}$$

EXEMPLE 5.3.3:

Prenons $\mathcal{X} = \mathcal{Y} = \{0,1,2\}$, et $b(0) = b(1) = 1$, $b(2) = 4$ et les probabilités de transition suivantes :

$$Q = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Alors $\beta_{min} = 1$ et $C_{min} = 1$.

Soit X une source test qui atteint $C(\beta)$, notons $\alpha_i = P(X = i)$. On a alors

$$\begin{aligned} C(\beta) &= H(Y) - H(Y|X) \\ &= H(Y) \qquad \text{car Y = X} \\ &= H(X) \\ &= H(\alpha_0, \alpha_1, \alpha_2) \end{aligned}$$

Cette quantité est maximale pour $\alpha_0 = \alpha_1 = \alpha_2 = 1/3$, et on obtient ainsi $C_{max} = \log_2 3$ et $\beta_{max} = 2$.

En maximisant la quantité $H(\alpha_0, \alpha_1, \alpha_2)$ sous la contrainte $1 \cdot \alpha_0 + 1 \cdot \alpha_1 + 4 \cdot \alpha_2 = \beta$, on obtient finalement

$$\forall \beta \in [1,2], C(\beta) = H\left(2/3 - \beta/6, 2/3 - \beta/6, \beta/3 - 1/3\right).$$

5.4 Réciproque du théorème de codage de canal avec coût

Nous reprenons les calculs de la Section 5.2 en prenant en compte les contraintes de coût.

Le message M est uniforme sur $[1, 2^{nR}]$ et encodé dans des mots-code β-admissibles : $b[X_i^{(n)}] \leq n\beta$ pour tout $i = 1, \ldots 2^{nR}$.

On a alors

$$\begin{aligned} nR &= H(M) \\ &\leq 1 + nR\overline{P_E} + \sum_{i=1}^{n} I(X_i, Y_i) \\ &\leq 1 + nR\overline{P_E} + \sum_{i=1}^{n} C(E[b(X_i)]) \\ &\leq 1 + nR\overline{P_E} + nC\left(\frac{1}{n}\sum_{i=1}^{n} E[b(X_i)]\right) \\ &\leq 1 + nR\overline{P_E} + nC(\beta), \end{aligned}$$

où les deux dernières inégalités découlent de la concavité de C et de sa monotonie an β. On a donc

$$R \leq \frac{C(\beta)}{1 - \overline{P_E}} + \frac{1}{n(1 - \overline{P_E})},$$

et donc si $\overline{P_E} \to 0$ alors $R \leq C(\beta)$.

5.5 Le théorème de codage de canal

On sait déjà que l'on ne trouvera pas de code permettant de transmettre avec une probabilité d'erreur moyenne arbitrairement faible si $R > C$. Le théorème suivant donne un résultat positif (mais non constructif) dans le cas $R < C$.

Théorème 5.5.1 *Pour un canal discret sans mémoire de fonction capacité-coût $C(\beta)$, pour tout $\beta_0 \geq \beta_{min}$, et des réels $\beta > \beta_0$, $R < C(\beta_0)$, $\epsilon > 0$, pour n suffisamment grand, il existe un code $C \stackrel{def}{=} \{\underline{x}_1, \ldots, \underline{x}_M\}$ de longueur n et une règle de décodage, tel que :*

(a) chaque mot code $\underline{x}_i$ est β-admissible

(b) $M \geq 2^{\lceil Rn \rceil}$

(c) $P_E^{(i)} < \epsilon$ pour tout $i = 1, 2, \ldots, M$.

Démonstration. La preuve se décompose en deux parties. On va d'abord définir une fonction de décodage f pour un (M, n) code arbitraire. Puis on va montrer

qu'il existe un code satisfaisant aux propriétés $(a), (b), (c)$ en utilisant un argument probabiliste.
Soit $R < R' < C(\beta_0)$ et $\mathcal{T} = \{(\underline{x}, \underline{y}) / \, I(\underline{x}; \underline{y}) \geq nR'\} \subseteq \Omega$ où

$$I(\underline{x}; \underline{y}) = log_2 \frac{p(\underline{y}|\underline{x})}{p(\underline{y})} \text{ avec, } p(\underline{y}) = \sum_{\underline{x}} p(\underline{x}, \underline{y}) = \sum_{\underline{x}} p(\underline{x}) p(\underline{x}|\underline{y}).$$

On définit encore

$$B = \{\underline{x} \, / b(\underline{x}) \leq n\beta\}$$

$$\mathcal{T}^\star = \{(\underline{x}, \underline{y}) \in \mathcal{T} / \underline{x} \in B\}$$

Pour un code $C = \{\underline{x}_1, \ldots, \underline{x}_M\}$ de longueur n, on définit la règle de décodage : pour chaque $\underline{y}$, on pose $S(\underline{y})$ l'ensemble des « bons candidats »

$$S(\underline{y}) = \{\underline{x} / (\underline{x}, \underline{y}) \in \mathcal{T}^\star\} \subseteq B$$

Lorsque l'on reçoit $\underline{y}$, si $S(\underline{y})$ est réduit à un élément $\underline{x}_i$, alors on pose $f(\underline{y}) = \underline{x}_i$ sinon on pose $f(\underline{y}) = ?$
Pour le code C et avec cette règle de décodage, si $\underline{x}_i$ est transmis et $\underline{y}$ est reçu, on a :

$$P_E^i \leq P(\underline{x}_i \notin S(\underline{y})) + \sum_{j \neq i} P(\underline{x}_j \in S(\underline{y}))$$

Définissons les fonctions indicatrices Δ et $\overline{\Delta}$ comme suit

$$\Delta(\underline{x}, \underline{y}) = \begin{cases} 1, & \text{si } (\underline{x}, \underline{y}) \in \mathcal{T}^\star \\ 0, & \text{sinon} \end{cases} \quad \text{et} \quad \overline{\Delta} = 1 - \Delta$$

Cela permet de réecrire la majoration sur P_E^i de manière explicite comme une fonction (déterministe) du code $C = \{\underline{x}_1, \ldots, \underline{x}_M\}$:

$$P_E^i \leq \sum_{\underline{y}} \overline{\Delta}(\underline{x}_i, \underline{y}) p(\underline{y}|\underline{x}_i) + \sum_{j \neq i} \sum_{\underline{y}} \Delta(\underline{x}_j, \underline{y}) p(\underline{y}|\underline{x}_i) \stackrel{def}{=} Q_i(\underline{x}_1, \ldots, \underline{x}_M)$$

Voici maintenant la deuxième partie de la preuve (où l'on montre l'existence). L'idée du « random coding » est de définir une distribution de probabilités sur C, sous laquelle l'espérance des Q_i -ce sont alors des v.a- tende vers 0, pour $M = 2^{\lceil Rn \rceil}$. L'argument sera du type : « si en moyenne une quantité est petite, alors il existe

des tirages qui la rendent effectivement petite ». On fait le choix suivant pour la distribution de probabilité sur les codes C :

$$p(\underline{x}_1, \ldots, \underline{x}_M) = \prod_{i=1}^{M} p(\underline{x}_i)$$

où $p(\underline{x}_i) = \prod_{k=1}^{n} p(x_{ik})$ atteint $C(\beta_0)$.

Remarque 5.5.1 – *l'intuition est la suivante : la loi $p(\underline{x})$ en entrée du canal permet de maximiser l'information mutuelle entre la sortie du canal et l'entrée, ce qui 'signifie' que les mots code $\underline{x}$ ayant une 'forte' probabilité $p(\underline{x})$ sont bien transmis. La loi choisie sur les codes met plus de 'poids' sur les mots code transmis de manière fiable et donc la 'densité' de tels mots code est plusélevée.*

– *Nous avons défini des codes jusqu'à présent comme des ensembles. Ici, la définition d'un code est un M-uplet. En particulier, il peut y avoir deux mot-codes identiques qui vont donc systématiquement donner une erreur avec la règle de décodage que nous avons définie. Pour des raisons de symmétries, il est cependant plus facile d'effectuer les calculs avec cette distribution et de modifier le code pour résoudre ces problèmes a posteriori, ce que nous ferons à la fin de la preuve.*

En prenant la moyenne par rapport à la loi sur les codes, on obtient :

$$\mathbb{E}[Q_i] = \underbrace{\mathbb{E}\big[\sum_{\underline{y}} \overline{\Delta}(\underline{x}_i, \underline{y}) p(\underline{y}|\underline{x}_i)\big]}_{E_1} + \sum_{j \neq i} \underbrace{\mathbb{E}\big[\sum_{\underline{y}} \Delta(\underline{x}_j, \underline{y}) p(\underline{y}|\underline{x}_i)\big]}_{E_2^{(j)}}$$

Il s'agit à présent de majorer E_1 et $E_2^{(j)}$ par des quantités tendant vers 0.

$$\begin{aligned} E_1 &= \sum_{\underline{x}_1, \ldots, \underline{x}_M} p(\underline{x}_1 \ldots p(\underline{x}_M) \sum_{\underline{y}} \overline{\Delta}(\underline{x}_i, \underline{y}) p(\underline{y}|\underline{x}_i) \\ &= \sum_{\underline{x}, \underline{y}} p(\underline{x}) p(\underline{y}|\underline{x}) \overline{\Delta}(\underline{x}_i, \underline{y}) \\ &= P((\underline{x}, \underline{y}) \notin \mathcal{T}^\star) \\ &\leq P(\underline{x}, \underline{y}) \notin \mathcal{T}) + P(\underline{x} \notin B) \end{aligned}$$

Ce qui se réecrit

$$E_1 \leq P(I(\underline{x}, \underline{y}) < nR') + P(b(\underline{x}) > n\beta)$$

dans l'espace probabilisé $\Omega \stackrel{def}{=} \mathcal{X}^n \times \mathcal{Y}^n = \{(\underline{x}, \underline{y})/\underline{x} \in \mathcal{X}^n, \underline{y} \in \mathcal{Y}^n\}$, muni de la mesure de probabilité :

$$p(\underline{x}, \underline{y}) = p(\underline{x})p(\underline{y}|\underline{x})$$

où

- $p(\underline{x}) = p(x_1) \dots p(x_n)$ et $p(.)$ atteint la capacité $c(\beta_0)$
- $p(\underline{y}|\underline{x}) = \prod_{i=1}^n p(y_i|x_i)$

Mais $b(\underline{x}) = \sum_{k=1}^n b(x_k)$ est une somme de v.a i.i.d de moyenne $\mathbb{E}[b(X)] \leq \beta_0 < \beta$. Par la Loi Faible des Grands Nombres (LFGN)

$$\lim_{n\to\infty} P(b(\underline{x}) > n\beta) = 0$$

Par la propriété sans mémoire du canal, on peut écrire

$$I(\underline{x}, \underline{y}) = \sum_{k=1}^n \log \frac{p(y_k|x_k)}{p(y_k)} = \sum_{k=1}^n I(x_k, y_k)$$

Donc sous la probabilité $p(\underline{x}, \underline{y})$, nous avons à nouveau une somme de v.a. i.i.d de moyenne :

$$\mathbb{E}[I(x_k, y_k)] = I(X, Y) = C(\beta_0) > R'$$

Et on peut conclure que $P(I(\underline{x}, \underline{y}) < nR') \to 0$ par la LFGN. Passons à $E_2^{(j)}$

$$\begin{aligned} E_2^{(j)} &= \sum_{\underline{x}_j, \underline{y}} p(\underline{x}_j)\Delta(\underline{x}_j, \underline{y}) \sum_{\underline{x}_i} p(\underline{x}_i)p(\underline{y}|\underline{x}_i) \\ &= \sum_{\underline{x}, \underline{y}} p(\underline{x})\Delta(\underline{x}, \underline{y})p(\underline{y}) \\ &\leq \sum_{\underline{x}, \underline{y} \in \mathcal{T}} p(\underline{x})p(\underline{y}) \end{aligned}$$

Par définition de $\mathcal{T}$, $p(\underline{x})p(\underline{y}) \leq p(\underline{x}, \underline{y})2^{-R'n}$. Donc

$$E_2^{(j)} \leq 2^{-R'n} \sum_{\underline{x}, \underline{y} \in \mathcal{T}} p(\underline{x}, \underline{y}) \leq 2^{-R'n}$$

et $\sum_{j \neq i} E_2^{(j)} \leq M2^{-R'n}$. Pour $M = 2^{\lceil Rn \rceil + 1}$ (ce choix de M paraît arbitraire mais va être justifié dans la suite) et $R' > R$, de dernier majorant tend vers 0 quand n tend vers l'infini.

Donc pour n assez grand, $\mathbb{E}[Q_i] \leq \epsilon$ avec $M = 22^{\lceil Rn \rceil + 1}$. On définit l'erreur moyenne par :

$$P_E(C) = P_E(\underline{x}_1, \ldots, \underline{x}_M) = \frac{1}{M} \sum_{i=1}^{M} P_E^i(\underline{x}_1, \ldots, \underline{x}_M)$$

Pour $M = 2^{\lceil Rn \rceil + 1}$ et n grand, la majoration de chacun des $\mathbb{E}[Q_i]$ donne :

$$\mathbb{E}[P_E] < \epsilon.$$

Donc en particulier il existe un code $\{\underline{x}_1, \ldots, \underline{x}_M\}$ avec probabilité d'erreur moyenne

$$P_E(\underline{x}_1, \ldots, \underline{x}_M) < \epsilon$$

Attention : il peut cependant exister des mots code $\underline{x}_i$ avec

$$b(\underline{x}_i) > n\beta \text{ ou } P_E^i > \epsilon.$$

Pour régler ce problème on procède de la façon suivante. On sait qu'au plus la moitié des mots codes $\underline{x}_i$ sont tels que $P_E^{(i)} \geq \epsilon$ sinon $P_E \geq \epsilon/2$ (ce qui fournit une contradiction quitte à changer $\epsilon \leftarrow \frac{\epsilon}{2}$ partout).
Si on supprime les mots code ayant une probabilité d'erreur $P_E^i \geq \epsilon$, on obtient un nouveau code avec $M \geq 2^{\lceil Rn \rceil}$ mots code. De plus le fait de supprimer des mots code ne peut faire que diminuer la probabilité d'erreur des mots code restant. Donc pour chacun des mots code restant, on a $P_E^i < \epsilon$.
De plus, automatiquement, si $b(\underline{x}_i) > n\beta$ alors $\underline{x}_i$ n'est pas dans le code obtenu. En effet, pour tout $\underline{y}$, comme $S(\underline{y}) = \{\underline{x}, (\underline{x}, \underline{y}) \in \mathcal{T} \,\&\, b(\underline{x}) \in B\}$ alors $\underline{x}_i \notin S(\underline{y})$ pour tout $\underline{y}$ et donc $P_E^{(i)} = 1$.
Finalement le code obtenu vérifie les trois propriétés attendues. □

5.6 Exercice : Canal avec feedback

Dans un canal avec feedback, au moment de l'encodage du $(n+1)$-ème bit, on sait quels ont été les n premiers bits reçus. Le but est de montrer qu'un tel canal a la même capacité que s'il n'y avait pas de feedback. Clairement $C_{\text{feedback}} \geq C$.
Un $(2^{nR}, n)$ code avec feedback est une suite de fonctions $\{1, \ldots, 2^{nR}\} \times \mathcal{Y}^{i-1} \to \mathcal{X}$ et une fonction de décodage $g : \mathcal{Y}^n \to \{1, \ldots, 2^{nR}\} \cup \{?\}$.
Si W est uniformément distribuée dans $\{1, \ldots, 2^{nR}\}$

$$P_E^{(n)} = P(g(Y^{(n)}) \neq W)$$

donc

$$\begin{aligned} nR = H(W) &= H(W|\hat{W}) + I(W;\hat{W}) \\ \text{(Fano)} &\leq 1 + P(W \neq \hat{W})nR + I(W;\hat{W}) \\ \text{(Data Processing Ineq.)} &\leq 1 + P_E^{(n)}nR + I(W;Y^{(n)}) \end{aligned}$$

et

$$\begin{aligned} I(W;Y^{(n)}) &= H(Y^{(n)}) - H(Y^{(n)}|W) \\ &= H(Y^{(n)}) - \sum_{i=1}^{n} H(Y_i|Y_1,\ldots,Y_{i-1},W) \\ &\leq \sum_i H(Y_i) - \sum_i H(Y_i|X_i) \\ &= \sum_{i=1}^{n} I(X_i,Y_i) \leq nC, \end{aligned}$$

où la première inégalité vient du fait que X_i est une fonctions des $Y_1,\ldots,Y_{i-1},W$ donc $H(Y_i|Y_1,\ldots,Y_{i-1},W) = H(Y_i|Y_1,\ldots,Y_{i-1},W,X_i)$ et que sachant X_i, Y_i est indépendante des $Y_1,\ldots,Y_{i-1},W$ donc $H(Y_i|Y_1,\ldots,Y_{i-1},W,X_i) = H(Y_i|X_i)$.

Donc $nR \leq 1 + P_E^{(n)}nR + nC$ soit

$$R \leq \frac{C}{1 - P_E^{(n)}} + \frac{1}{n(1 - P_E^{(n)}}$$

Donc pour tout $(2^{nR}, n)$ code avec feedback tel que $P_E^{(n)} \to 0$ quand $n \to \infty$, on doit avoir $R \leq C$, ce qui suffit à conclure en vertu du théorème de codage de canal.

Canal à effacement

On cherche à déterminer la capacité C de ce canal.

$$C = \max I(X;Y) = \max_{p(x)} H(Y) - H(\alpha)$$

Posons $\pi := P(X = 1)$, alors $H(Y)$ se réecrit

$$H(Y) = H((1-\pi)\alpha, \alpha, \pi(1-\alpha)) = H(\alpha) + (1-\alpha)H(\pi)$$

et donc $C = 1 - \alpha$.
En fait, on savait qu'on ne pourrait pas faire mieux grâce au canal analogue avec feedback. Cependant pour le canal à effacement avec feedback, la méthode de

transmission suivante est naturelle : le symbole reçu est transmis dans le canal feednback, donc si e a été reçu, on peut réemettre jusqu'à transmission du bit voulu. Dans ce cas on pourra transmettre au taux $1-\alpha$ car la transmission d'un bit prend un temps aléatoire suivant une loi géométrique de paramètre de moyenne $\frac{1}{1-\alpha}$. Donc pour transmettre k bits, il faut de l'ordre de $n = \frac{k}{1-\alpha}$ utilisations du canal (lorsque k est grand).

Chapitre 6

Sources sans mémoire et leurs fonctions taux-distorsion

6.1 La fonction taux-distorsion

Une source (discrète) émet une suite de symboles u dans un ensemble fini $\mathcal{U}$ appelé l'alphabet de la source. Une source sans mémoire est caractérisée par sa statistique $p(u)$: la suite de symboles est une suite de v.a. U_i indépendantes et identiquement distribuées (i.i.d.) de loi p.

Dans ce chapitre, nous considérons un encodage dans un alphabet de destination $\mathcal{V}$. Pour tout couple $(u, v) \in \mathcal{U} \times \mathcal{V}$, $d(u, v) \geq 0$ est l'erreur ou distortion associée. Pour des vecteurs $(\underline{u}, \underline{v}) \in \mathcal{U}^k \times \mathcal{V}^k$, on note $d(\underline{u}, \underline{v}) = \sum_{i=1}^{k} d(u_i, v_i)$.

Nous prendrons la convention : $\mathcal{U} = \{0, \ldots, r-1\}$ et $\mathcal{V} = \{0, \ldots, s-1\}$. La matrice de distorsion D est la matrice $r \times s$ à coefficients $d(u, v)$.

Étant donné $k > 0$, une source et une fonction d'encodage, la distorsion moyenne est alors :

$$E(d) \quad = \quad E[d(\underline{U}, \underline{V})] = \sum_{\underline{u}, \underline{v}} p(\underline{u}) p(\underline{v}|\underline{u}) d(\underline{u}, \underline{v}).$$

On définit alors la fonction

$$R_k(\delta) = \min \{ I(U; V),\ E(d) \leq k\delta \},$$

où la minimisation est faite sur toutes les paires $(\underline{U}, \underline{V})$ telles que $\underline{U} = (U_1, \ldots U_k)$ et les $U_1, \ldots U_k$ sont i.i.d. de loi $p(u)$.

Remarque 6.1.1 – *$p(\underline{v}|\underline{u})$ peut être vue comme la probabilité de transition d'un canal. On parlera alors de canal test.*
– *la fonction $p(\underline{v}|\underline{u}) \mapsto I(U;V)$ est continue donc elle atteint son minimum sur un compact.*
– *soit $\delta_{\min} = \sum_{u\in\mathcal{U}} p(u) \min_v d(v,u)$. On a $E(d) \geq k\delta_{\min}$, donc $R_k(\delta)$ est définie pour $\delta \geq \delta_{\min}$.*
– *la fonction $\delta \mapsto R_k(\delta)$ est clairement décroissante en δ.*

Définition 6.1.1 *La fonction taux-distorsion de la source est*

$$R(\delta) = \inf_k \frac{1}{k} R_k(\delta).$$

Théorème 6.1.1 *La fonction $\delta \mapsto R_k(\delta)$ est convexe pour $\delta \geq \delta_{\min}$.*

Démonstration. Soit $\alpha_1, \alpha_2 \geq 0$ avec $\alpha_1 + \alpha_2 = 1$. L'inégalité $R_k(\alpha_1\delta_1 + \alpha_2\delta_2) \leq \alpha_1 R_k(\delta_1) + \alpha_2 R_k(\delta_2)$ s'obtient facilement en considérant les probabilités de transition $p(\underline{v}|\underline{u}) = \alpha_1 p_1(\underline{v}|\underline{u}) + \alpha_2 p_2(\underline{v}|\underline{u})$ où p_i est un canal test atteignant $R_k(\delta_i)$ pour $i = 1, 2$ et en utilisant la convexité de $I(\underline{U};\underline{V})$. □

Théorème 6.1.2 *Pour une source discrète sans mémoire $R_k(\delta) = kR_1(\delta)$ pour tout k et $\delta \geq \delta_{\min}$.*

Démonstration. Soit $p(\underline{v}|\underline{u})$ une probabilité de transition atteignant $R_k(\delta)$, c'est à dire telle que

$$I(\underline{U};\underline{V}) = R_k(\delta) \text{ et } E[d(\underline{U},\underline{V})] \leq k\delta.$$

Comme les $U_1, U_2, \ldots, U_k$ sont indépendants, on a $I(\underline{U};\underline{V}) \geq \sum_{i=1}^k I(U_i;V_i)$. Avec la notation $\delta_i = E[d(U_i,V_i)]$, on a $I(U_i;V_i) \geq R_1(\delta_i)$ donc $I(\underline{U};\underline{V}) \geq \sum_{i=1}^k R_1(\delta_i)$. Par le résultat de convexité démontré ci-dessus, on obtient :

$$\sum_{i=1}^k R_1(\delta_i) \geq kR_1\left(\frac{1}{k}\sum_{i=1}^k \delta_i\right) \geq kR_1(\delta),$$

où on utilise la monotonie de R_1 dans la dernière inégalité. On a donc $R_k(\delta) \geq kR_1(\delta)$. Pour démontrer l'inégalité inverse, il suffit de considérer $p(\underline{v}|\underline{u}) = \prod_{i=1}^k p(v_i|u_i)$ où $p(v|u)$ atteint $R_1(\delta)$. □

Corollaire 6.1.1 *Pour une source discrète sans mémoire $R(\delta) = R_1(\delta) = \min\{I(U;V),\ E[d] \leq$*

Propriétés de $R(\delta)$ pour une source DSM :

- la fonction $\delta \mapsto R(\delta)$ est décroissante, convexe donc continue pour $\delta > \delta_{\min}$. Elle est également continue en $\delta_{\min}$ (exo !).
- Soit $\delta_{\max} = \min_v \sum_u p(u)d(u,v)$ alors $R(\delta) = 0$ si et seulement si $\delta \geq \delta_{\max}$. En effet, soit $v^* = \arg\min \sum_u p(u)d(u,v)$ alors l'encodage déterministe $u \to v^*$ est tel que $I(U;v^*) = 0$ et $E[d] = \delta_{\max}$. Donc $R(\delta) = 0$ pour $\delta \geq \delta_{\max}$. Inversement si $R(\delta) = 0$ alors le canal test doit avoir U et V indépendants et donc
$$E[d] = \sum p(u)p(v)d(u,v) \geq \sum_v p(v)\delta_{\max} = \delta_{\max}.$$
- Comme $R(\delta)$ est décroissante, convexe pour $\delta \geq \delta_{\min}$ et constante pour $\delta \geq \delta_{\max}$, $R(\delta)$ est strictement décroissante pour $\delta_{\min} \leq \delta \leq \delta_{\max}$. En particulier, on a
$$R(\delta) = \min\{I(U;V),\ E[d] = \delta\} \text{ pour } \delta_{\min} \leq \delta \leq \delta_{\max}.$$
- Dans le cas particulier où chaque ligne de D a au moins un 0 et chaque colonne au plus un 0, on a $\delta_{\min} = 0$ et $E[d] = 0$ ssi $u \to v \in v(u) = \{v,\ d(u,v) = 0\}$ qui sont des ensembles disjoints par notre hypothèse. Donc $R(0) = I(U;V) = H(U) - H(U|V) = H(U)$.

EXEMPLE 6.1.1:

$\mathcal{U} = \mathcal{V} = \{0,1\}$ avec $P(0) = p = 1 - P(1) \leq 1/2$ et matrice de distorsion
$$D = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$
On a $\delta_{\min} = 0$, $\delta_{\max} = \min\{p, 1-p\} = p$ et $R(0) = H(p)$. Pour $0 < \delta < p$, soit (U,V) atteignant $R(\delta)$. On a $I(U,V) = H(U) - H(U|V) = H(p) - H(U|V)$ et $E[d] = P(U \neq V) = \delta$. Donc par l'inégalité de Fano, on a $H(U|V) \leq H(\delta)$ et donc $R(\delta \geq H(p) - H(\delta)$. Pour obtenir l'inégalité opposée, il faut trouver (U,V) tel que $E[d] = \delta$ et $I(U;V) = H(p) - H(\delta)$. Un candidat est de choisir $p(u|v) = \delta \mathbf{1}(u \neq v) + (1-\delta)\mathbf{1}(u = v)$ puisqu'on a alors $E[d] = \delta$ et $H(U|V) = H(\delta)$. Il faut donc vérifier qu'il est bien possible de choisir $\alpha = P(V = 0)$ tel que $P(U = 0) = p$. On calcule que $p = \alpha(1-\delta) + (1-\alpha)\delta$ soit $\alpha = \frac{p-\delta}{1-2\delta} \in [0,1]$ car $0 < \delta < p \leq 1/2$. Au final, on a montrer que
$$R(\delta) = \begin{cases} H(p) - H(\delta) & 0 \leq \delta \leq p, \\ 0 & \delta \geq p. \end{cases}$$

6.2 Théorème de codage de source de Shannon

Soit $U^{(k)} = (U_1, \ldots, U_k)$ un vecteur representant les k premiers symboles émis par une source. On suppose que ces k symboles sont 'compressés'en n bits $X^{(n)} = (X_1, \ldots, X_n)$ et qu'il est possible à partir de ces n bits de les 'décoder' en k symboles de l'alphabet de destination : $V^{(k)} = (V_1, \ldots, V_k)$ de telle sorte que $\sum_{i=1}^{k} E[d(U_i, V_i)] \leq k\delta$. Il est facile de voir que l'on a nécessairement :

$$\frac{n}{k} \geq R(\delta). \tag{6.1}$$

En effet, par définition, on a $I(U^{(k)}; V^{(k)}) \geq R_k(\delta)$. De plus par l'inégalité data processing, on a $I(U^{(k)}; V^{(k)}) \leq I(X^{(k)}; V^{(k)}) \leq H(X^{(k)}) \leq n$, ce qui implique que $R_k(\delta)/k \leq n/k$ et donc (6.1) découle (sans avoir fait l'hypothèse que la source est sans mémoire).

On voit donc qu'il faut au moins $R(\delta)$ bits pour représenter un symbole source si la distorsion moyenne doit être inférieure à δ. Nous allons maintenant montrer qu'il ne faut 'pas plus de $R(\delta)$ bits'.

Définition 6.2.1 *Un* **code de source** *de longueur k est un ensemble $C = \{\underline{v}_1, \ldots, \underline{v}_M\} \subset \mathcal{V}^k$. Le taux du code est $R = \frac{1}{k} \log_2 M$.*

Pour chaque suite de la source $\underline{u} = (u_1, \ldots, u_k)$, $f(\underline{u})$ est le mot code $\underline{v}_i$ le plus proche de $\underline{u}$:

$$d(\underline{u}, f(\underline{u})) \leq d(\underline{u}, \underline{v}_j) \quad \forall j \in \{1, \ldots, M\}.$$

La distorsion moyenne de C est $d(C) = \frac{1}{k} \sum_{\underline{u} \in \mathcal{U}^k} p(\underline{u}) d(\underline{u}, f(\underline{u}))$, avec $p(\underline{u}) = p(u_1) \ldots p(u_k)$.

EXEMPLE 6.2.1: CODE $(7, 4)$ DE HAMMING : $\mathcal{U} = \mathcal{V} = \{0, 1\}$ avec $p(0) = p(1) = 1/2$ et $D = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$. Chacun des 128 vecteurs binaires diffère d'au plus un bit par rapport à un mot code donc

$$d(C) = \frac{1}{7} \left(\frac{128 - 16}{128} \right) = \frac{1}{8}.$$

Théorème 6.2.1 *Soit $\delta \geq \delta_{\min}$. Pour tout $\delta' > \delta$ et $R' > R(\delta)$ pour k suffisament grand, il existe un codage de source C de longueur k avec M mots code tel que*

- *a) $M \leq 2^{\lfloor kR' \rfloor}$;*

– *b)* $D(C) < \delta'$.

Démonstration. Soient $R(\delta) < R'' < R'$ et $\delta < \delta'' < \delta'$. □

Pour un code $C = \{\underline{v}_1, \ldots, \underline{v}_M\}$ on définit les ensembles :

$$\begin{aligned} S &= \{\underline{u},\ d(\underline{u}, f(\underline{u})) \leq k\delta''\} \text{ (suites bien représentées par } C), \\ T &= \{\underline{u},\ d(\underline{u}, f(\underline{u})) > k\delta''\} \text{ (suites mal représentées par } C). \end{aligned}$$

On a donc

$$d(C) \leq \underbrace{\frac{1}{k}\sum_{\underline{u}\in S} p(\underline{u}) d(\underline{u}, f(\underline{u}))}_{\leq \delta''} + \underbrace{\frac{1}{k}\sum_{\underline{u}\in T} p(\underline{u}) d(\underline{u}, f(\underline{u}))}_{\leq B\sum_{\underline{u}\in T} p(\underline{u})}$$

avec $B = \max d(u, v)$. On a donc

$$d(C) \leq \delta'' + BP\left(d(\underline{u}, f(\underline{u})) > k\delta''\right), \tag{6.2}$$

où la probabilité vient de l'aléa de la source. Pour être plus explicite, on définit

$$\Delta(\underline{u}, \underline{v}) = \begin{cases} 1 & \text{si } d(\underline{u}, \underline{v}) \leq k\delta'' \\ 0 & \text{sinon.} \end{cases}$$

On peut donc écrire (6.2) comme suit :

$$d(C) \leq \delta'' + B \underbrace{\sum_{\underline{u}} p(\underline{u}) \prod_{i=1}^{M} \left(1 - \Delta(\underline{u}, \underline{v}_i)\right)}_{K(C)}.$$

Il faut donc trouver un code de source C de longueur k avec au plus $2^{\lfloor kR' \rfloor}$ mots code et tel que $K(C) < \frac{\delta' - \delta''}{B}$. Nous utilisons à nouveau un argument de **random coding**. Soit $p(u, v)$ la probabilité sur $\mathcal{U} \times \mathcal{V}$ qui atteint $R(\delta) : I(U; V) = R(\delta)$ et $E[d(U, V)] \leq \delta$. On note les marginales par $p(u) = \sum_v p(u, v)$ et $p(v) = \sum_u p(u, v)$. On définit alors une probabilité sur $\mathcal{U}^k \times \mathcal{V}^k$ par $p(\underline{u}) = \prod_{i=1}^k p(u_i)$, $p(\underline{v}|\underline{u}) = \prod_{i=1}^k p(v_i|u_i)$ de telle sorte que $p(\underline{u}, \underline{v}) = \prod_{i=1}^k p(u_i, v_i)$. On définit alors une probabilité sur les codes de longueur k et ayant M mots code (vu comme des M-uplets de vecteurs de dimensions k) par :

$$p(C) = \prod_{i=1}^{M} p(\underline{v}_i) \text{ avec } p(\underline{v}_i) = \prod_{j=1}^{k} p(v_{ij}).$$

Nous calculons la moyenne de $K(C)$ sous cette probabilité :

$$\begin{aligned} E[K] &= \sum_{\underline{v}_1 \cdots \underline{v}_M} p(\underline{v}_1) \ldots p(\underline{v}_M) \sum_{\underline{u}} p(\underline{u}) \prod_{i=1}^{M} (1 - \Delta(\underline{u}, \underline{v}_i)) \\ &= \sum_{\underline{u}} p(\underline{u}) \left(\sum_{\underline{v}} p(\underline{v})(1 - \Delta(\underline{u}, \underline{v}) \right)^M . \end{aligned}$$

On définit maintenant

$$\Delta_0(\underline{u}, \underline{v}) = \begin{cases} 1 & \text{si } d(\underline{u}, \underline{v}) \leq k\delta'' \text{ et } I(\underline{u}; \underline{v}) \leq kR'', \\ 0 & \text{sinon.} \end{cases}$$

avec $I(\underline{u}; \underline{v}) = \log_2 \left(\frac{p(\underline{v}|\underline{u})}{p(\underline{v})} \right)$. Si $\delta_0(\underline{u}, \underline{v}) = 1$ alors $p(\underline{v}) \geq 2^{-kR''} p(\underline{v}|\underline{u})$ et donc

$$E[K] \leq \sum_{\underline{u}} p(\underline{u}) \left(1 - 2^{-kR''} \sum_{\underline{v}} p(\underline{v}|\underline{u}) \Delta_0(\underline{u}, \underline{v}) \right)^M ,$$

et donc en utilisant l'inégalité $(1 - xy)^M \leq 1 - x + e^{-yM}$ valide pour $0 \leq x, y \leq 1$ et $M > 0$ avec $x = \sum_{\underline{v}} p(\underline{v}|\underline{u}) \Delta_0(\underline{u}, \underline{v})$ et $y = 2^{-kR''}$, on obtient

$$\begin{aligned} E[k] &\leq 1 - \sum_{\underline{u}, \underline{v}} p(\underline{u}, \underline{v}) \Delta_0(\underline{u}, \underline{v}) + \exp\left(-2^{-kR''} M \right) \\ &\leq P(d(\underline{U}, \underline{V}) > k\delta'') + P(I(\underline{U}; \underline{V}) > kR'') + \exp\left(-2^{-kR''} M \right) . \end{aligned}$$

Le dernier terme tend vers 0 quand k tend vers l'infini car $M = 2^{\lfloor kR' \rfloor}$ avec $R' > R''$. Le fait que les deux premiers temes tendent également vers 0 découle de la loi faible des grands nombres. En effet, il suffit d'écrire : $d(\underline{U}, \underline{V}) = \sum_{i=1}^{k} d(U_i, v_i)$ qui est une somme de k v.a. i.i.d. de moyenne $E[d(U, V)] \leq \delta < \delta''$. De même $I(\underline{U}; \underline{V}) = \sum_{i=1}^{k} I(U_i; V_i)$ est une somme de v.a. i.i.d. de moyenne $R(\delta) < R''$.

Chapitre 7

Le théorème de codage source-canal

Un tel système est caractérisé par :
- une v.a. $\underline{U}$ modélisant la source ;
- une fonction d'encodage modélisé par $p(\underline{X}|\underline{U})$;
- une probabilité de transition pour le canal $p(\underline{Y}|\underline{X})$;
- une fonction de décodage $p(\underline{V}|\underline{Y})$.

$$\begin{aligned} \textbf{Coût moyen} : \ \overline{\beta} &= \frac{1}{n} E[b(\underline{X})] \\ \textbf{Distorsion moyenne} : \ \overline{\delta} &= \frac{1}{k} E[d(\underline{U}, \underline{V})] \\ \textbf{Taux de transmission} : \ \overline{r} &= \frac{k}{n} \end{aligned}$$

Le but est d'avoir des $\overline{\beta}$ et $\overline{\delta}$ aussi petits que possible tandis que $\overline{r}$ est aussi grand que possible.

Théorème 7.0.2 *Pour une source et un canal donné :*

1. *Les paramètres $\overline{\beta}$, $\overline{\delta}$, $\overline{r}$ doivent satisfaire $\overline{r} \leq \frac{C(\overline{\beta})}{R(\overline{\delta})}$*
2. *Inversement étant donné $\beta > \beta_{min}$, $\delta > \delta_{min}$ et $r < \frac{C(\beta)}{R(\delta)}$, il est possible de construire une système tel que décrit ci-dessus avec $\overline{\beta} \leq \beta$, $\overline{\delta} \leq \delta$ et $\overline{r} \geq r$*

Démonstration.

1. La suite $(\underline{U}, \underline{X}, \underline{Y}, \underline{V})$ constitue une chaîne de Markov, donc $I(\underline{U};\underline{V}) \leq I(\underline{X};\underline{Y})$. De plus $E[b(\underline{X})] = n\overline{\beta}$ implique que $I(\underline{X};\underline{Y})) \leq C_n(\overline{\beta})$ et comme $C_n(\overline{\beta}) \leq nC(\overline{\beta})$, on a au final : $I(\underline{X};\underline{Y}) \leq nC(\overline{\beta})$. De plus $E[d(\underline{U},\underline{V})] = k\overline{\delta}$ implies that $I(\underline{U};\underline{(V)}) \geq kR(\overline{\delta})$ et donc le point (a) est vérifié.
2. On vérifie que l'on peut choisir :

$$\begin{aligned} \beta_{min} \leq \beta_0 < \beta, \\ \delta_{min} \leq \delta_0 < \delta_1 < \delta \\ C' < C(\beta_0), \\ R' > R(\delta_0), \\ r < \frac{C'}{R'}. \end{aligned}$$

Théorème de codage de source

Pour k_0 suffisamment grand, il existe un code source C de longueur k_0 avec M_1 mots code tel que $M_1 \leq 2^{\lfloor R'k_0 \rfloor}$ et $d(C) = \frac{1}{k_0}E[d_{min}(\underline{U})] < \delta_1$, avec $d_{min}(\underline{U}) = min\{d(\underline{U}, \underline{V_i}), \underline{V_i} \in C\}$.
Pour m défini plus tard, on note $k = k_0 m$. L'encodeur de source partitionne $\underline{U} = (U_1, \ldots, U_k)$ en m blocs de longueur k_0 et émet m mots code de source correspondant à ces blocs, $\underline{W} = (W_1, \ldots, W_k)$ est une suite de m mots code de C, le nombre de valeurs distinctes possibles pour $\underline{W}$ est inférieur ou égal à $M_1^m \leq 2^{k_0 m R'}$.

Encodeur de canal

On définit la distorsion pire cas du code C pour $\underline{u} \in \mathcal{U}^{k_0}$ par $d_{max}(\underline{u}) = max\{\ d(\underline{u}, \underline{v_i}),\ \underline{v_i} \in C\ \}$, et la distorsion pire cas du code par $D(C) = \frac{1}{k_0}E[d_{max}(\underline{U})]$ où l'espérance est par rapport a la statistique de la source.
On note $\epsilon = \frac{\delta - \delta_1}{D(C)}$ et pour chaque $m = 1, 2, \ldots$ soit $n_m = \lceil mk_0R'/c' \rceil$.

Théorème de codage de canal

Pour m suffisament grand, il existe un code $\{\underline{x_1}, \ldots, \underline{x}_{M_2}\}$ de longueur n_m et une règle de décodage tel que :

$$\begin{aligned} b(\underline{X_i}) &\leq n_m\beta \\ M_2 &\geq 2^{\lceil C'n_m \rceil} \geq 2^{mk_0R'} \\ P_E^{(i)} &< \epsilon. \end{aligned}$$

On suppose de plus que m est suffisament grand pour que $\overline{r} = \frac{k}{n_m} = \frac{k_0 m}{\lceil m k_0 \frac{R'}{C'} \rceil} \geq r$, ce qui est possiblie puisque $r < \frac{C'}{R'}$. On a également $n\overline{\beta} = E[b(\underline{X})] \leq n\beta$.
Il reste donc à prouver que $\overline{\delta} \leq \delta$.
Le décodeur de canal est celui du théorème de codage de canal, le décodeur de source est celui dont la sortie $\underline{V} = (V_1, \ldots, V_k)$ de m mots code de source si possible, sinon erreur.

$$B = \begin{cases} 0 & \text{si } \underline{Z} = \underline{X} \\ 1 & \text{sinon.} \end{cases}$$

$$E[d(\underline{U}, \underline{V})] = E[d(\underline{U}, \underline{V}) \mid B = 0\ ].P(B = 0) + E[d(\underline{U}, \underline{V}) \mid B = 1].P(B = 1)$$

$$\text{Si } B = 0,\ d(\underline{U}, \underline{V}) = \sum_{l=0}^{m} d_{min}(\underline{U}^l) \text{ avec } \underline{U} = (U^1, \ldots, U^m) \text{ blocs de taille } k_0,$$

$$\text{alors } E[d(\underline{U}, \underline{V}) \mid B = 0] = m.E[d_{min}(\underline{U})] < k_0 \delta_1 m$$

$$\text{Si } B = 1,\ E[d(\underline{U}, \underline{V}) \mid B = 1] \leq m.E[d_{max}(\underline{U}) \mid B = 1]$$

$$E[d_{max}(\underline{U}) \mid B = 1] = \sum_{i=1}^{M_2} E[d_{max}(\underline{U}) \mid B = 1, \underline{X} = \underline{X_i}].P[\underline{X} = \underline{X_i} \mid B = 1]$$

$$= \sum_{i=1}^{M_2} E[d_{max}(\underline{U}) \mid \underline{X} = \underline{X_i}].P_E^{(i)}.\frac{P(\underline{X} = \underline{X_i})}{P(B = 1)}$$

$$E[d_{max}(U)|B = 1].P(B = 1) \leq m\epsilon.E[d_{max}(\underline{U})] = k_0 m(\delta - \delta_1) = k(\delta - \delta_1)$$

Au final, on a $\overline{\delta} = k^{-1} E\left[d(\underline{U}; \underline{V})\right] < \delta$.

□

Chapitre 8

Codes linéaires

8.1 Décodage par maximum de vraisemblance

Etant donné un canal et un code, on cherche un décodage minimisant la probabilité d'erreur, c'est à dire une fonction $y_j \mapsto x_j^*$ pour $j = 1, \ldots, L$.

On a

$$1 - P_e = \sum_{j=1}^{L} p(y_j) p(x_j^* | y_j),$$

où $p(y_j)$ ne dépend pas du décodage. On doit donc choisir x_j^* qui maximise $p(x|y_j)$.

De manière similaire, si w est envoyé et v est reçu, il faut maximiser $p(w|v) = \frac{p(w) \prod_i p(v_i|w_i)}{p(v)}$ (dans le cas d'un canal sans mémoire).

Si tous les M symboles en entrée sont équiprobables alors le décodage optimal quand y est reçu est x_i qui maximise $p(y|x_i)$ car :

$$p(x_i|y) = \frac{p(x_i)p(y|x_i)}{p(y)} = \frac{1}{Mp(y)} p(y|x_i).$$

C'est le décodage par maximum de vraisemblance.

Considérons maintenant un canal binaire symétrique (de probabilité d'erreur $\epsilon \in (0, 1/2)$ ainsi qu'un code binaire de longueur n. Si tous les mots code sont équiprobables, alors le décodage par maximum de vraisemblance consiste à maximiser

$$p(v|w) = \epsilon^{d(w,v)} (1 - \epsilon)^{n - d(w,v)},$$

où $d(w, v)$ est la distance de Hamming entre le mot reçu v et le mot code w. Pour un mot reçu fixé v, on note $d_i = d(w_i, v)$ la distance de Hamming entre le i-ème mot code et v. On a alors :

$$\frac{p(v|w_1)}{p(v|w_2)} = \left(\frac{1-\epsilon}{\epsilon}\right)^{d_2-d_1}.$$

Donc pour $0 < \epsilon < 1/2$, on a $\frac{1-\epsilon}{\epsilon} > 1$ et donc $p(v|w_1) > p(v|w_2)$ si et seulement si $d_1 < d_2$.

On a donc pour un canal binaire symétrique et un code dont tous les mots code sont équiprobables : le décodage minimisant la probabilité d'erreur consiste à minimiser la distance de Hamming, i.e. choisir le mot code le plus proche pour la distance de Hamming du mot reçu. Si il y a plusieurs mots code à distance minimale, la probabilité d'erreur ne dépend pas du choix du mot code.

8.2 Géométrie de Hamming

Nous formalisons la notion de bon code ayant des mots code élpoignés.

La distance de Hamming pour des mots non nécessairement binaires est $d_H(\underline{x}, \underline{y}) = w_H(\underline{x} - \underline{y}) = \sum_i \mathbf{1}(x_i \neq y_i)$
Soit $C = \{\underline{x}_1, \ldots, \underline{x}_M\}$ un code de longueur n.
On veut que C soit capable de corriger les erreurs de poids $\leq e$.
On envoie $\underline{x}_i$, et $\underline{y} = \underline{x}_i + \underline{z}$ est reçu ; si $w_H(\underline{z}) \leq e$ alors $\hat{\underline{x}}_i = \underline{x}$. Le code est capable de corriger les erreurs de poids $\leq e$ ssi la distance entre chaque pair de mots code est supérieur ou égale à $2e + 1$, i.e. $d_H(\underline{x}_i, \underline{x}_j) \geq 2e + 1 \ \forall i, j$
Soit $d_{min}(C) = min(d_H(\underline{x}, \underline{x}', x \neq x' \ \underline{x}, \underline{x}' \in C))$ la distance minimale du code C.

Théorème 8.2.1 *Un code $C = \{\underline{x}_1, \ldots, \underline{x}_M\}$ est capable de corriger toutes les erreurs de poids $\leq e$ ssi $d_{min}(C) \geq 2e + 1$*

Théorème 8.2.2 *Si un code sur un alphabet à q lettres et de longueur n est constitué de s mots code et corrige toutes les erreurs de poids $\leq e$ alors*

$$s \leq \frac{q^n}{\sum_{i=0}^{e} \binom{n}{i}(q-1)^i}$$

Démonstration. Toutes les boules $\{v, d_H(w_i, v) \leq e\}$ doivent être disjointes et le cardinal de ces boules est $\sum_{i=0}^{e} \binom{n}{i}(q-1)^i$ d'où l'inégalité. □

Cette condition n'es pas suffisante comme montré par l'exemple suivnat : prendre en binaire, i.e $q = 2$, $e = 1$ et $n = 4$ de telle sorte que $\lfloor 2^n/(n+1) = 16/5 \rfloor = 3$. Cependant il n'existe pas de code corrigeant une erreur avec plus de 2 mots code.

8.3 Codes linéaires

$\mathcal{X} = F_q = \{0, 1, \ldots, q-1\}$ avec q un nombre premier

Définition 8.3.1 *Un (n, k) code linéaire sur F_q est un sous-espace de dimention k de F_q^n, n est la longueur du code, k = dimension du code, taux = k/n.*

Un code est décrit par k mots code linéairement indépendant $\underline{x}_1, \ldots, \underline{x}_k$.
Chaque mot code et l'un des q^k combinaison linéaire

$$\sum_{i=1}^{k} \alpha_i \underline{x}_i, \quad \alpha_i \in F_q$$

Définition 8.3.2 *Soit C un (n, k) code linéaire sur F_q. Une matrice G dont l'espace engendré par les lignes est C est une matrice générant C.*

EXEMPLE 8.3.1:
(5, 1) code C_1, $G = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \end{pmatrix}$

EXEMPLE 8.3.2:
(7, 4) code

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

est le code de Hamming (7, 4)

Encodage (n, k) code linéaire a q^k mots code message $\underline{u} = (u_1, \ldots, u_k) \in F_q^k$ mot code $\underline{x} = \underline{u}G$.

Matrice de parité / parity check matrix
équation de parité $a_1x_1 + a_2x_2 + \cdots + a_nx_n = 0$ qui est satisfaite pour tout $\underline{x} \in C$ l'espace des vecteurs $\underline{a} = (a_1, \ldots, a_n)$ est l'espace $C^\perp$ appelé code dual de C.

$C^\perp$ a pour dimension $n-k$, $C^\perp$ est un (n, n-k) code. Une matrice de parité pour C est une matrice générant $C^\perp$

Définition 8.3.3 *C un (n,k) code linéaire sur F_q, une matrice H avec la propriété, $H\underline{x}^t = 0 \Leftrightarrow \underline{x}$ in$C$ est appellée matrice de parité de C*

Pour un code linéaire, $d_H(\underline{x}, \underline{x}') = w_H(\underline{x} - \underline{x}')$ et $\underline{x} - \underline{x}' \in C$
donc $d_{min}(C) = min\{w_H(\underline{x}), \underline{x} \in C, \underline{x} \neq 0\}$
Il suffit donc de calculer $q^k - 1$ poids au lieu des $q^k(q^k-1)/2$ distances entre les différents mots code.

Théorème 8.3.1 *Si C est un (n,k) code linéaire sur F_q, de matrice de parité H, $d_{min}(C) =$ le plus petite nombre de colonnes de H qui sont linéairement dépendantes.*

Démonstration.
$H\underline{x}^t = 0 = x_1\underline{c_1} + x_2\underline{c_2} + \cdots + x_n\underline{c_n}$ $\mathcal{Y} = F_q$. □

8.4 Codes de Hamming (binaires)

Définition 8.4.1 *H de taille $m \times 2^m - 1$ binaire telle que les colonnes de H sont les $2^m - 1$ vecteurs non nuls de F_2^m dans un ordre donné. Alors les $n = 2^m - 1$, $k = 2^m - 1 - m$ code linéaire sur F_2 de matrice de parité H est appelé un code de Hamming de longueur $2^m - 1$.*

D'après le théorème précédent, la distance minimale d'un code de Hamming est de 3. Un code de Hamming peut donc corriger une erreur.

De plus la boule de Hamming de rayon 1 contient en dimension n, exavtement $1 + n$ mots. Donc un code corrigeant une erreur a au plus (comme nous l'avons vu précédement) $2^n/(n+1)$ mots code différents. En prenant $n = 2^m - 1$ comme c'est le cas pour les codes de Hamming, on obtient : $2^{2^m-1}/2^m = 2^{2^m-m-1} = 2^k$. Donc les codes de Hamming sont des codes parfaits : les boules de rayon 1 autour des mots code forment une partition de F_2^n.

8.5 Décodage du syndrome

$\underline{x}$ est transmis, $\underline{y}$ est reçu, et on note $\underline{z} = \underline{y} - \underline{x}$ le motif d'erreur
Si $z_i \neq 0$ il y a erreur sur le $i^{\text{ème}}$ bit

$\underline{s} = H\underline{y}^t = H\underline{z}^t$ est appelé le syndrome et ne dépend que du motif d'erreur.
L'ensemble des solution en $\underline{z}$ de $H\underline{z}^t = \underline{s}$ forme un coset du code C du type : $C + \underline{z}_0 = \{\underline{x} + \underline{z}_0, \underline{x} \in C\}$

Il y a q^{n-k} cosets de C correspondants aux q^{n-k} syndromes possibles. Chaque coset contient exactement q^k éléments. Pour distinguer parmi les q^k éléments du coset celui correspondant au motif d'erruer, il faut faire des hypothèses sur le canal. Pour le canal binaire symétrique avec probabilité d'erreur $\epsilon < 1/2$, on choisit $\underline{z}$ de poids minimal dans le coset.

Dans le cas du code de Hamming, si le motif d'erreur a poids 1 alors le syndrome donne la colonne de H correspondant à l'endroit de l'erruer, i.e. le décodage consiste a changer le bit correspondant.

Chapitre 9

Codes cycliques

9.1 Propriétés générales des codes cycliques

Définition 9.1.1 *Un code (n, k) linéaire sur F_q est cyclique si pour tout mot code $\underline{C} = (C_0, \ldots, C_{n-1})$ le shift droit de $\underline{C}$: $\underline{C}^R = (C_{n-1}, C_0, \ldots, C_{n-2})$ est aussi un mot code.*

Exemple 9.1.1:

Les codes à répétition.

Exemple 9.1.2:

Le $(7, 3)$-code sur F_2 de matrice :

$$G = \begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}$$

en effet :

$$L_1 \overset{C^R}{\to} L_2 \overset{C^R}{\to} L_3 \overset{C^R}{\to} L_1 + L_2 \overset{C^R}{\to} L_1 + L_2 + L_3 \overset{C^R}{\to} L_1 + L_3 \overset{C^R}{\to} L_2 + L_3 \overset{C^R}{\to} L_1$$

EXEMPLE 9.1.3:

Le $(4,2)$-code sur F_3 de matrice :

$$G = \begin{pmatrix} 1 & 0 & 2 & 0 \\ 1 & 1 & 2 & 2 \end{pmatrix}$$

en effet :

$$L_1 \overset{C^R}{\to} 2L_1 + L_2 \overset{C^R}{\to} 2L_1 \overset{C^R}{\to} L_1 + 2L_2 \overset{C^R}{\to} L_1$$

et

$$L_2 \overset{C^R}{\to} L_1 + L_2 \overset{C^R}{\to} 2L_2 \overset{C^R}{\to} 2L_1 + 2L_2 \overset{C^R}{\to} L_2$$

Définition 9.1.2 *La fonction génératrice d'un mot code* $\underline{C} = (C_0, \ldots, C_{n-1})$ *est le polynôme* $C(X) = C_0 + \ldots + C_{n-1}X^{n-1}$

Théorème 9.1.1 *Si* $\underline{C} = (C_0, \ldots, C_{n-1})$ *est un mot code de fonction génératrice* $C(X)$ *alors la fonction génératrice de* $\underline{C}^R$ *est* $C^R(X) = XC(X) \mod (X^n - 1)$

Démonstration. On a par définition :

$$\begin{aligned} C(X) &= C_0 + \ldots C_{n-1}X^{n-1} \\ XC(X) &= XC_0 + \ldots C_{n-1}X^n \\ C^R(X) &= C_{n-1} + C_0X + \cdots + C_{n-2}X^{n-1}. \end{aligned}$$

Donc $XC(X) - C^R(X) = C_{n-1}(X^n - 1)$ et,

$XC(X) \mod (X^n - 1) = \left[C^R(X) \mod (X^n - 1)\right] + \left[XC(X) - C^R(X) \mod (X^n - 1)\right] = 0$

□

Notation : $[P(X)]_n = P(X) \mod (X^n - 1)$.
On a en particulier, $\forall i\ X^i \mod (X^n - 1) = [X^i]_n = X^{i \mod n}$

Théorème 9.1.2 *Si C est un code (n,k) cyclique et si $C(X)$ est un mot code (on identifie mot code et fonction génératrice d'un mot code) alors $\forall P(X)$ $[P(X)C(X)]_n$ est aussi un mot code.*

Démonstration. Découle du théorème précédent et de la linéarité du code. □

Définition 9.1.3 *Une fonction génératrice de degré minimal dans C est appelée generatrice du code C.*

EXEMPLE 9.1.4:

Reprenons l'exemple 9.1.2
$C_1(X) = 1 + X^2 + X^3 + X^4$ est un polynôme générateur de C.

EXEMPLE 9.1.5:

Reprenons l'exemple 9.1.3
$C_1(X) = 1 + 2X^2$ est un polynôme générateur de C.
Remarquons que les polynômes générateurs ne sont pas uniques $2C_1(X) = 2 + X^2$ est aussi polynôme générateur.

Le lemme suivant montre qu'il existe un unique polynôme générateur unitaire.

Lemme 9.1.1 *Soit C un code cyclique de générateur $g(X)$*

1. *Si $\tilde{g}(X)$ est un autre polynôme générateur alors $\tilde{g}(X) = \lambda g(X)$ avec $\lambda \in F_q$*
2. *Si $P(X)$ est un polynôme tel que $[P(X)]_n$ est un mot code alors $g \mid P$*

Démonstration.
2. : On a $P(x) = Q(x)g(x) + R(x)$ or $[P(X)]_n$ et $[Q(X)g(X)]_n$ sont des mots codes donc $R(X)$ est un mot code de degré strictement inférieur à g c'est donc 0.
1. : découle directement de 2. □

Théorème 9.1.3 *1. Si C est un code (n,k) cyclique sur F_q alors son générateur g est un diviseur de $X^n - 1$ de plus $\underline{C} = (C_0, \ldots, C_{n-1})$ est un mot code ssi il est divisible par g et $k = n - \deg g$.*

2. *Inversement si g est un diviseur de $X^n - 1$ alors il existe un (n, k) code cyclique de générateur g avec $k = n - \deg g$. C'est exactement l'ensemble des vecteurs $(C_0, \ldots, C_{n-1})$ de fonction génératrice divisible par g.*

Démonstration.

1. : Soit $P(X) = X^n - 1$ alors $[P(X)]_n = 0$ donc est un mot code par conséquent le lemme nous donne que $g \mid P$.
$\Longleftarrow$ immédiat d'après le théorème précédent car g est un mot code.
$\Longrightarrow$ Immédiat d'après le lemme précédent car si $\underline{C}$ mot code alors $C(X) = [C(X)]_n$ donc $C(X) = I(X)g(X)$ avec $\deg I \leq n - 1 - \deg g$.

2. : Soit $g(x) \mid X^n - 1$. $C(X)$ est un multiple de g ssi $C(X) = I(X)g(X)$ et il est facile de voire que le code est un code (n, k)-linéaire.
Pour montrer que le code est cyclique on montre que $[XI(X)g(X)]_n$ est un mot code. Or

$$\begin{aligned} [xI(x)g(x)]_n \mod g(x) &= xI(x)g(x) \mod (x^n - 1) \mod g(x) \\ &= xI(x)g(x) \mod g(x) \text{ car } g(X) \mid X^n - 1 \\ &= 0 \end{aligned}$$

□

Définition 9.1.4 *On appelle polynôme de parité d'un code C le polynôme $h(X) = \frac{X^n - 1}{g(X)}$ avec g générateur de C.*

Corollaire 9.1.1 *Soit C un code (n, k) cyclique de générateur g et de polynôme de parité h.*
On définit :

$$G_1 = \begin{pmatrix} g_0 & \ldots & g_r & 0 & \ldots & 0 \\ 0 & \ddots & & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & & \ddots & 0 \\ 0 & \ldots & 0 & g_0 & \ldots & g_r \end{pmatrix} = \begin{pmatrix} g(X) \\ Xg(X) \\ \vdots \\ X^{k-1}g(X) \end{pmatrix}$$

et

$$H_1 = \begin{pmatrix} h_k & \ldots & h_0 & 0 & \ldots & 0 \\ 0 & \ddots & & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & & \ddots & 0 \\ 0 & \ldots & 0 & h_k & \ldots & h_0 \end{pmatrix}$$

La matrice G_1 génère le code C et sa matrice de parité est donnée par H_1. De plus le vecteur $\underline{I} = (I_0, \ldots, I_{k-1})$ est encodé par $\underline{C} = \underline{I}G_1$ i.e. $C(X) = I(X)g(X)$.

Démonstration. le produit scalaire de la i-ème ligne de G et de la j-ème ligne de H est le coefficient de x^{k-i+j} du produit $h(x)g(x)$. Cependant par définition $h(x)g(x) = x^n - 1$ et comme $k - i + j \in [1, n-1]$, ce coefficient est nul. □

9.2 Classification des codes binaires cycliques de longueur 7

On a :

$$\begin{aligned} X^7 - 1 &= (1 + X^2 + X^3 + X^4)(1 + X^2 + X^3) \\ &= (1 + X)(1 + X + X^3)(1 + X^2 + X^3) \end{aligned}$$

d'où :

(n, k)	$g(X)$	
$(7, 7)$	1	
$(7, 6)$	$X + 1$	
$(7, 4)$	$X^3 + X + 1$	code de Hamming
$(7, 4)$	$X^3 + X^2 + 1$	code de Hamming
$(7, 3)$	$(X + 1)(X^3 + X + 1)$	Exemple 9.1.2
$(7, 3)$	$(X + 1)(X^3 + X^2 + 1)$	Exemple 9.1.2 à l'envers
$(7, 1)$	$(X^3 + X^2 + 1)(X^3 + X + 1)$	code à répétition
$(7, 0)$	$X^7 - 1$	

Remarque : Le polynôme $g(x) = x^3 + x + 1$ divise $x^n - 1$ pour tout n multiple de 7. Il est donc possible de construire des codes $(7, 4), (14, 11), (21, 18), \ldots$ cependant tous ces codes sauf le premier contiennent un vecteur de fonction génératrice $x^7 - 1$ donc ont une distance minimum de égale à 2 et ne peuvent pas être utilisés pour corriger des erreurs.

Définition 9.2.1 *Le plus petit entier n tel que $g(x)|x^n - 1$ est la période de $g(x)$ (car c'est la période de la suite $\{x^i \mod g(x)\}_{i \geq 0}$).*