Licence de Sciences Economiques et de Gestion, Mention Mass

Théorie de l'information

Avertissement : ce polycopié n'entend pas se substituer au cours et n'a pour autre but que d'être une aide éventuelle ; il n'est notamment pas garanti sans erreur ni exhaustif. De plus ce cours est optionnel et étant destiné aux étudiants de l'UFR de Sciences Economiques et de Gestion il constitue une introduction à la théorie de l'information, ainsi de nombreuses notions sont introduites sans justification

Joëlle Cohen

2 février 2007

Table des matières

Chapitre 1

Introduction

1.1 Fax

Soit un document A4 à transmettre de format $21 \times 29,7$ cm, soit une surface de $623,7$ cm^2. On transmet des points à imprimer par le codage suivant : 1 = point noir et 0 = point blanc. On imprime 6200 points par cm^2 donc en tout $6200 \times 623,7 = 3866948$ bits pour un document A4.

Si on utilise un débit de 14,4 Kbits par seconde alors une page A4 nécessite 268 s soit 4 min 28 s pour être transmis.

Grâce à des techniques de codage on peut réduire ce temps à une vingtaine de secondes.

1.2 Séquence vidéo sur CD-Rom

L'élément le plus élémentaire d'une image (pixel) est caractérisé par 3 composantes RVB en analogique et est codé numériquement selon 3 composantes Y, C_R et C_B.

Y est la luminance et C_R et C_B représentent la chrominance.

Chacune de ces 3 composantes est codée sur 8 bits : on a donc $2^8 = 256$ valeurs distinctes pour Y, pour C_R et pour C_B.

Soit une image composée de 480 lignes à 720 pixels par ligne. Par combien de bits sera-t-elle codée ? On tient compte du fait que l'oeil humain ne distingue pas la chrominance de 2 pixels contigüs donc le codage de la chrominance se fera à raison de 8 bits soit un octet pour 2 pixels.

Par conséquent une telle image sera codée par $1 \times 720 \times 480(\frac{1}{2} + \frac{1}{2}) + 720 \times 480 = 691200$ octets.

Si on a une vidéo à 25 images par seconde alors une seconde d'images nécessite $25 \times 691200 = 17530000 = 17,53$ Moctets.

Un CD-Rom contient 650 Mo donc peut stocker $\frac{650}{17,53} = 37s$ de vidéo.

Avec un codage MPEG1 on atteint plus d'une heure d'image.

1.3 Fichier MP3

Un signal analogique stéréo a deux voies : une voie de gauche et une voie de droite chacune de fréquence comprise entre 0 et 20 Khz.

Le signal analogique est quasiment continu dans le temps. Mais le passage au numérique suppose alors de rendre ce signal continu discret. On va donc coder ponctuellement un certain nombre de sons par seconde : on appelle cela un échantillonage.

Pour numériser une tel signal en qualité CD Audio, on requiert une fréquence d'échantillonage de 44,1Khz (c'est-à-dire on code 44100 échantillons de son par seconde) à raison de 16 bits par échantillon. Donc une seconde de son sera codé par

$$\underbrace{44100}_{\text{échantillons}} \times \underbrace{16}_{\text{bits par échantillons}} \times \underbrace{2}_{\text{voies}} = 1,411 \text{ Mbits/seconde}$$

Sans codage plus performant un CD-Rom contient alors $650 \times 8/1,411 = 3685$ secondes soit environ 1h de musique.

Le codage MP3 (MPEG 1 layer 3) permet de coder une seconde de musique par 128 Kbits.

Donc une minute de musique nécessite $\dfrac{128000 \times 60}{8} = 0,96$ Moctets.

Un CD-Rom peut alors contenir environ 650 minutes soit plus de 10h de musique.

1.4 Qu'est-ce-que la théorie de l'information

La théorie de l'information est une théorie mathématique qui décrit les aspects fondamentaux des systèmes de communication. Elle a été initiée par C. Shannon dans les années 1940.

Un système de communication est la transmission d'une information depuis une source à travers un canal jusqu'à un récepteur.

Une source peut être
- une voix
- une suite de symboles binaires (bits)
- un signal électromagnétique ...

Le canal peut être
- une ligne téléphonique
- une liaison radio
- un support optique
- un support magnétique ...

Pour améliorer la transmission, on code la source pour réduire le débit de la source : cela peut se faire avec ou sans perte d'informations (on se limitera à sans perte) et consiste à transmettre en moyenne moins de symboles qu'il n'en provient de la source.

Le canal de transmission est sujet à diverses perturbations dues à l'environnement que l'on nommera bruit. Pour contrer ces perturbations qui peuvent engendrer soit perte soit déformation de l'information, on utilisera un codage de canal qui, contrairement au précédent, ajoutera des informations au message à transmettre ce qui augmentera le débit nécessaire.

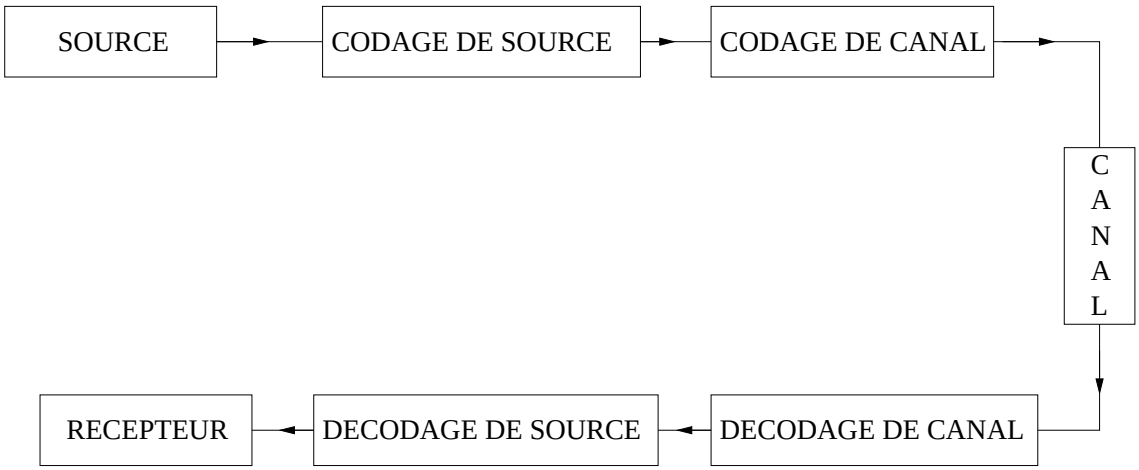

FIG. 1.1 – schéma de transmission d'une information

Bien sûr, à la réception il faut décoder ce qui arrive du canal pour ainsi restituer le premier codage de l'information transmise qui à son tour sera décodé pour arriver au récepteur (voir figure 1.1).

Chapitre 2

Mesure de l'information

Un message reçu n'apporte de l'information que si son contenu n'est pas connu à l'avance de son destinataire. Par exemple, si je connais le prochain bit à recevoir, je n'ai pas besoin de le recevoir.

On va supposer que l'ensemble de tous les messages possibles est fini. Alors fournir une information c'est lever l'incertitude à l'issue d'une expérience aléatoire.

Cette incertitude peut varier pour un même événement si on a connaissance d'une autre information : pour 2 événements E et F, si

- $p(E/F) < p(E)$ alors l'incertitude sur E augmente si on sait que F s'est réalisé
- $p(E/F) = p(E)$ alors E et F sont indépendants, l'information apportée par F n'influence pas l'incertitude sur la survenue de E
- $p(E/F) > p(E)$ alors E devient plus probable si on sait que F s'est réalisé

L'idée de Shannon est de quantifier cette donnée sachant que plus le contenu du message est rare plus l'information apportée est importante. A contrario, si on est sûr de recevoir un certain message il n'apporte aucune information et la mesure de l'information apportée devra alors être nulle.

On voit alors qu'il y a un lien entre la probabilité de recevoir une information et la mesure que l'on veut en donner : ce lien que l'on cherche à établir doit respecter les idées ci-dessus.

De plus on souhaite que la quantité d'information apportée par 2 événements indépendants soit la somme des quantités d'information apportées par chacun.

Rappel : si E et F sont 2 événements la probabilité conditionnelle est égale à

$$p(E/F) = \frac{p(E \cap F)}{p(F)}$$

E et F sont indépendants *si et seulement si* $p(E \cap F) = p(E)p(F)$ ce qui équivaut à

$$p(E/F) = p(E)$$

2.1 Quantité d'information

Définition 2.1 *Soit E un événement. On appelle quantité d'information de E la valeur*

$$I(E) = -\log_2 p(E) = -\frac{\ln p(E)}{\ln 2}$$

où $p(E)$ est la probabilité de E.

On remarque que la fonction I vérifie bien les requis exprimés plus haut : si $p(E)$ diminue, $I(E)$ augmente et si $p(E) = 1$ alors $I(E) = 0$.

Le choix du logarithme en base 2 n'est pas anodin : définissons le *bit* (binary unit) comme la quantité d'information apportée par le choix entre deux valeurs équiprobables.

Donc, si on a une variable E qui prend deux valeurs équiprobables (par exemple pile ou face pour une pièce non truquée) alors la quantité d'information apportée par la réalisation de $\{E = pile\}$ est de 1 bit par définition du bit. Et on a bien $1 = -\log_2 \frac{1}{2}$.

L'unité de quantité d'information est le bit.

Pour représenter une information de n bits, il faut alors n symboles binaires.

Par exemple, si on 16 valeurs possibles équiprobables, alors une valeur a une quantité d'information égal à 4 et il faut 4 bits (binary digit) pour représenter toutes les valeurs. Mais ce ne sera pas toujours le cas si la distribution de probabilité est inégale.

On montre maintenant que cette définition répond à l'additivité requise pour I.

Propriété 2.1 *Si E et F sont 2 événements indépendants alors $I(E \cap F) = I(E) + I(F)$. La quantité d'information apportées par 2 événements indépendants est la somme de leur quantités d'information respectives.*

preuve : $I(E \cap F) = -\log_2 p(E \cap F) = -\log_2 p(E)p(F) = -\log_2 p(E) - \log_2 p(F) = I(E) + I(F)$ ■

***exemple* 2.1** : soit un jeu de 32 cartes dans lequel on effectue des tirages et les événements $E = \{$la carte tirée est un valet de cœur$\}$ et $F = \{$la carte tirée est un cœur$\}$

On a pour E, $p(E) = 1/32$ et $I(E) = 5$, et pour F, $p(F) = 1/4$ et $I(F) = 2$.

E et F ne sont pas indépendants car $p(E/F) = \dfrac{p(E \cap F)}{p(F)} = \dfrac{1/32}{1/4} = 1/8$

□

Cela nous mène à définir l'information mutuelle pour 2 événements.

2.2 Information mutuelle

On veut mesurer l'apport d'information de l'événement F sur l'événement E. Si la réalisation de F augment la probabilité de réalisation de E on veut que cette mesure soit positive et inversement si F augmente l'incertitude sur E cette mesure doit être négative. Enfin si les deux événements sont indépendants cette mesure doit être nulle.

Définition 2.2 *Soient E et F 2 événements. L'information apportée par F sur E est défini par*

$$I(F \to E) = \log_2 \frac{p(E/F)}{p(E)}$$

Contrairement à la quantité d'information, l'information mutuelle n'est pas toujours un réel positif.

Propriété 2.2 *Soient E et F 2 événements.*

$$I(F \to E) = I(E \to F) = \log_2 \frac{p(E \cap F)}{p(E)p(F)}$$

On notera alors $I(F \to E) = I(E, F) = I(F, E)$ et on l'appellera information mutuelle entre E et F.

preuve : par définition, $p(E/F) = \dfrac{p(E \cap F)}{p(F)}$ donc $\dfrac{p(E/F)}{p(E)} = \dfrac{p(E \cap F)}{p(E)p(F)} = \dfrac{p(F/E)}{p(F)}$ d'où $I(E, F) = I(F, E) = \log_2 \dfrac{p(E \cap F)}{p(E)p(F)}$ ■

On remarque que si
- $I(E, F) > 0$ alors la réalisation d'un des 2 événements augmente la probabilité de l'autre (diminue son incertitude)
- $I(E, F) = 0$ alors E et F sont indépendants, l'information mutuelle est nulle
- $I(E, F) < 0$ alors la réalisation d'un des 2 événements diminue la probabilité de l'autre (augmente son incertitude)
- $p(E \cap F) = 0$ alors la réalisation d'un des 2 événements rend impossible la réalisation de l'autre et $I(E, F) = -\infty$

La propriété suivante établit un lien entre la quantité d'information et l'information mutuelle.

Propriété 2.3 $I(E \cap F) = I(E) + I(F) - I(E, F)$

preuve : $I(E \cap F) = -\log_2 p(E \cap F)$ et d'après la proposition 2.2 on a
$I(E, F) = \log_2 p(E \cap F) - \log_2 p(E) - \log_2 p(F) = \log_2 p(E \cap F) + I(E) + I(F)$ donc
$-\log_2 p(E \cap F) = I(E) + I(F) - I(E, F)$ ■

2.3 Entropie

2.3.1 entropie d'une variable aléatoire

Prenons l'exemple d'un dé. On voudrait connaître comme contenu d'information la valeur du dé après un lancer. Soit alors X la variable aléatoire à valeurs dans $\{1, 2, 3, 4, 5, 6\}$. X peut prendre 6 valeurs et si le dé n'est pas truqué, les valeur sont équiprobables. Donc à chaque valeur correspond une quantité d'information de 2,58 bits ($= -\log_2 \frac{1}{6}$).

Mais supposons maintenant que le dé soit truqué et que la valeur 6 sorte avec une probabilité 0,5 et que les autres valeurs soient équiprobables. La quantité d'information pour chaque valeur n'est pas la même et pour avoir une vision globale on peut être intéressé à connaître l'information moyenne soit l'espérance de $I(X)$.

Elle vaut ici, $-\frac{1}{2}\log_2\frac{1}{2} - 5\times(\frac{1}{10}\log_2\frac{1}{10}) = \frac{1}{2}+\frac{1}{2}\times\log_2 10$ =1,22 bits.

Définition 2.3 *On appelle entropie de X l'espérance de $I(X)$ notée H(X).*

$H(X) = \sum_x p(X=x)I(X=x) = -\sum_x p(X=x)\log_2 p(X=x)$

- $H(X)$ est un réel positif comme $I(X=x)$.
- $H(X)$ correspond au nombre moyen d'éléments binaires pour coder les différentes valeurs de X.
- $H(X)$ n'est fonction que de la loi de probabilité de X, pas des valeurs prises par X.

***exemple* 2.2** : pour un jeu de 32 cartes, on définit la variable aléatoire X par $X=0$ si la carte est rouge, $X=1$ si la carte est un pique et $X=2$ si la carte est un trèfle. On a alors $H(X) = -(\frac{1}{2}\log_2\frac{1}{2}+\frac{1}{4}\log_2\frac{1}{4}+\frac{1}{4}\log_2\frac{1}{4}) = \frac{1}{2}+2\frac{1}{4}+2\frac{1}{4} = 1,5$ *bit*

□

Le résultat suivant aura pour conséquence de pouvoir mesurer l'efficacité d'un code

Théorème 2.4 *$H(X) \le \log_2 n$ si X prend n valeurs.*

$H(X) = \log_2 n$ si et seulement si X a une loi uniforme

(c'est-à-dire $p(X=x) = 1/n$ pour tout x).

preuve :

$$\begin{aligned} H(X) - \log_2 n &= -\sum_x p(X=x)\log_2 p(X=x) - \log_2 n(\sum_x p(X=x)) \\ &= -\sum_x p(X=x)(\log_2 p(X=x) + \log_2 n) \\ &= \sum_x p(X=x)\log_2\frac{1}{np(X=x)} \end{aligned}$$

Or on sait que $\ln x \le x-1$ donc $\log_2 x \le \frac{x-1}{\ln 2}$. D'où

$$\begin{aligned} H(X) - \log_2 n &\le \sum_x p(X=x)(\frac{1}{n\ln 2 p(X=x)} - \frac{1}{\ln 2}) \\ &\le \sum_x \frac{1}{n\ln 2} - \frac{1}{\ln 2}\sum_x p(X=x) \\ &\le \frac{1}{\ln 2} - \frac{1}{\ln 2} \\ &\le 0 \end{aligned}$$

car X prend n valeurs et donc $\sum_x \frac{1}{n \ln 2} = n\frac{1}{n \ln 2} = \frac{1}{\ln 2}$.

De plus si $p(X = x) = \frac{1}{n}$ pour tout x alors $H(X) = -\sum_x \frac{1}{n} \log_2 \frac{1}{n} = \sum_x \frac{1}{n} \log_2 n = \log_2 n$. ■

Conséquence : si X prend 2^r valeurs il faut en moyenne au maximum r symboles binaires pour représenter X. Intuitivement, on peut comprendre que pour déterminer une des 2^r valeurs, on aura besoin d'au plus r questions dont la réponse peut être oui ou non. Par exemple, si X représente la couleur d'une carte parmi $\{\spadesuit, \heartsuit, \diamondsuit, \clubsuit\}$, il suffit de demander noir ? puis selon la réponse ♠? ou ♡? pour connaître la couleur choisie.

Une autre propriété intéressante de l'entropie est la suivante :

Propriété 2.5 *L'entropie augmente lorsque le nombre de valeurs possibles augmente.*

preuve : Soit X une variable aléatoire prenant n valeurs $x_1, \ldots, x_n$ de probabilité respective $p_1, \ldots p_n$. Supposons que la valeur x_n soit partagée en deux valeurs y_n, z_n de probabilité respective p'_n, p''_n telles que $p'_n + p''_n = p_n, p'_n \neq 0, p''_n \neq 0$.

On a alors une nouvelle variable aléatoire X' dont l'entropie vaut
$H(X') = H(X) + p_n \log p_n - p'_n \log p'_n - p''_n \log p''_n$ donc
$H(X') - H(X) = (p'_n + p'_n) \log p_n - p'_n \log p'_n - p''_n \log p''_n = p'_n(\log p_n - \log p'_n) + p''_n(\log p_n - \log p''_n)$
or $\log p_n > \log p'_n$ et $\log p_n > \log p''_n$ donc $H(X') - H(X) > 0$. ■

2.3.2 entropie conditionnelle

Soient X, Y 2 variables aléatoires discrètes.

Définition 2.4 *On appelle entropie de X conditionnelle à $Y = y$*
$H(X/Y = y) = -\sum_x p(X = x/Y = y) \log_2 p(X = x/Y = y)$

On a alors

Définition 2.5 *On appelle entropie de X sachant Y*
$H(X/Y) = \sum_y p(Y = y) H(X/Y = y)$

Enfin on définit l'entropie mutuelle comme l'entropie d'un couple de variables aléatoires

Définition 2.6 *On appelle entropie mutuelle de X, Y*
$H(X, Y) = -\sum_{(x,y)} p(X = x, Y = y) \log_2 p(X = x, Y = y)$

Entropie de X sachant Y et entropie mutuelle sont deux valeurs positives. Le lien entre entropie mututelle et conditionnelle est donné par

Propriété 2.6 $H(X,Y) = H(X) + H(Y/X) = H(Y) + H(X/Y)$

preuve : $H(X,Y) = -\sum_{(x,y)} p(X=x, Y=y) \log_2 p(X=x, Y=y)$

or $p(X=x, Y=y) = p(Y=y/X=x)p(X=x)$ donc on a

$$\begin{aligned}
H(X,Y) &= -\sum_{(x,y)} p(Y=y/X=x)p(X=x)(\log_2 p(Y=y/X=x) + \log_2 p(X=x)) \\
&= -\sum_{(x,y)} p(Y=y/X=x)p(X=x) \log_2 p(Y=y/X=x) \\
&\quad -\sum_{(x,y)} p(Y=y/X=x)p(X=x) \log_2 p(X=x) \\
&= \sum_x p(X=x)[\sum_y -p(Y=y/X=x) \log_2 p(Y=y/X=x)] \\
&\quad -\sum_x [p(X=x) \log_2 p(X=x) \sum_y p(Y=y/X=x)] \\
&= \sum_x p(X=x)H(Y/X=x) - \sum_x [p(X=x) \log_2 p(X=x) \\
&= H(Y/X) + H(X)
\end{aligned}$$

■

Pour quantifier l'apport d'information à X fournie par Y, on mesure la différence entre l'entropie de X (l'information moyenne de X) et l'entropie conditionnelle de X sachant Y, soit $H(X) - H(X/Y)$. Il est facile de montrer que $H(X) - H(X/Y) = H(Y) - H(Y/X)$ c'est-à-dire ce que Y apporte à X est égal à ce que X peut apporter à Y.

En effet $H(X) - H(X/Y) = H(X,Y) - H(Y/X) - H(X/Y)$ d'après la proposition précédente. En l'appliquant de nouveau on déduit $H(X) - H(X/Y) = H(Y) - H(Y/X)$.

On va alors montrer que cette quantité est égale à l'espérance de $I(X=x, Y=y)$ défini plus haut comme information mutuelle.

$$\begin{aligned}
H(X) - H(X/Y) &= -\sum_x p(X=x) \log_2 p(X=x) - \sum_y p(Y=y)H(X/Y=y) \\
&= -\sum_x p(X=x) \log_2 p(X=x) \\
&\quad -\sum_y p(Y=y)(-\sum_x p(X=x/Y=y) \log_2 p(X=x/Y=y)) \\
&= \sum_x (-p(X=x) \log_2 p(X=x) + \sum_y p(Y=y)p(X=x/Y=y) \log_2 p(X=x/Y=y)
\end{aligned}$$

Or $p(X = x/Y = y) = \dfrac{p(X = x, Y = y)}{p(Y = y)}$ donc on a

$$\begin{aligned} H(X) - H(X/Y) &= \sum_x [-p(X = x) \log_2 p(X = x) \\ &\quad + \sum_y p(X = x, Y = y) \log_2 p(X = x, Y = y) - \sum_y p(X = x, Y = y) \log_2 p(Y = y) \end{aligned}$$

Mais $-\sum_x p(X = x) \log_2 p(X = x) = -\sum_x \sum_y p(X = x, Y = y) \log_2 p(X = x)$, d'où

$$\begin{aligned} H(X) - H(X/Y) &= \sum_x \sum_y [-p(X = x, Y = y) \log_2 p(X = x) \\ &\quad + p(X = x, Y = y) \log_2 p(X = x, Y = y) - p(X = x, Y = y) \log_2 p(Y = y) \\ &= \sum_x \sum_y p(X = x, Y = y) \log_2 \frac{p(X = x, Y = y)}{p(Y = y)p(X = x)} \\ &= I(X; Y) \end{aligned}$$

où $I(X; Y)$ est l'espérance de $I(X = x, Y = y)$. ■

On peut montrer que cette espérance est toujours positive contrairement à l'information mututelle.

Propriété 2.7 $I(X; Y) \geq 0$

preuve :

$$\begin{aligned} -I(X; Y) &= -\sum_x \sum_y p(X = x, Y = y) \log_2 \frac{p(X = x, Y = y)}{p(Y = y)p(X = x)} \\ &= \sum_x \sum_y p(X = x, Y = y) \log_2 \frac{p(Y = y)p(X = x)}{p(X = x, Y = y)} \\ &= \frac{1}{\ln 2} \sum_x \sum_y p(X = x, Y = y) \ln \frac{p(Y = y)p(X = x)}{p(X = x, Y = y)} \\ &\leq \frac{1}{\ln 2} \sum_x \sum_y p(X = x, Y = y) [\frac{p(Y = y)p(X = x)}{p(X = x, Y = y)} - 1] \\ &\leq \frac{1}{\ln 2} [\sum_x \sum_y p(X = x, Y = y) \frac{p(Y = y)p(X = x)}{p(X = x, Y = y)} - \sum_x \sum_y p(X = x, Y = y)] \\ &\leq \frac{1}{\ln 2} [\sum_x \sum_y p(Y = y)p(X = x) - 1] = 0 \end{aligned}$$

■

Remarque : si X et Y sont indépendants alors $I(X, Y) = 0$.

On déduit de ce résultat

Propriété 2.8 $H(X/Y) \leq H(X)$

Conséquence : le conditionnement diminue l'incertitude : on a besoin de moins de bit pour coder X en moyenne sachant Y.

Chapitre 3

Sources et codages de source

3.1 sources

Plusieurs modèles de sources peuvent être envisagés :
– sources discrètes sans mémoire
– sources discrètes stationnaires
– sources analogiques (onde caractérisée par un processus stochastique)...
On étudiera que les sources discrètes. Une source sera modélisée par une séquence aléatoire de lettres choisies dans un alphabet fini $A = \{a_1 \dots a_n\}$. Par exemple les données transmises par un ordinateur consttituent une source sur l'alphabet $\{0, 1\}$.

Définition 3.1 *Une source est stationnaire si et seulement si toutes les variables aléatoires à valeurs dans A ont la même loi de probabilité.*

Définition 3.2 *Une source a une mémoire d'ordre m si et seulement si* $p(X_k = a_k / \cap_{l=0}^{l=k-1} p(X_l = a_l) = p(X_k = a_k / \cap_{l=k-m}^{l=k-1} p(X_l = a_l))$.

Remarque : si $m = 1$ on a une chaîne de Markov et si $m = 0$ on a une source sans mémoire.

Dans le cas d'une source sans mémoire on peut mesurer l'entropie de la source par $H(A) = -\sum_{a \in A} p(a) \log_2 p(a)$.

3.2 Codages de sources discrètes sans mémoire

3.2.1 Exemple

Soit $A = \{a, b, c, d\}$ avec $p(a) = 1/2$, $p(b) = 1/4$, et $p(c) = p(d) = 1/8$. On propose de coder la source de 3 façons selon le tableau suivant

	C_1	C_2	C_3
a	00	000	1
b	01	01	01
c	10	10	001
d	11	1	000

La longueur moyenne pour coder une lettre sera

– $l(C_1) = 1/2 \times 2 + 1/4 \times 2 + 1/8 \times 2 + 1/8 \times 2 = 2$
– $l(C_2) = 1/2 \times 3 + 1/4 \times 2 + 1/8 \times 2 + 1/8 \times 1 = 2,375$
– $l(C_3) = 1/2 \times 1 + 1/4 \times 2 + 1/8 \times 3 + 1/8 \times 3 = 1,75$

C_3 est le meilleur des 3 codes c'est un comppression par rapport aux 2 bits (symboles binaires) nécessaires pour coder 2^2 valeurs.

Remarque : C_2 n'est pas un "bon code" parce qu'il ne permet pas un déchiffrage unique ; par exemple 101 = (1)(01)=(10)(1)

Les questions qui se posent suite à cet exemple sont les suivantes :
– peut-on toujours compresser sans perte d'information le contenu d'une source ?
– connaissant la source peut-on estimer le taux idéal de compression d'un codage ?
– comment faire un codage idéal ?

3.2.2 Exemple

Soit $A = \{a, b, c, d\}$ avec une loi uniforme. On montrera que C_1 est alors optimal.

Pour juger de l'efficacité d'un code on calcule pour un code donné la longueur moyenne de codage d'une lettre.

$l = (n_a + n_b + n_c + n_d)/4$ où n_a est le nombre de bits pour coder la lettre a,....

Si on veut que $l < 2$, alors il faut au moins un lettre codé par un seul symbole, par exemple $n_a = 1$, mais alors si b, c, d sont codés avec 2 symboles le décodage sera ambigü.

Par exemple,

a	0
b	10
c	11
d	01

le mot 010 peut être 0 10 soit *ab* ou bien 01 0 soit *da*. Donc on ne peut pas retenir cette solution.

De plus on doit s'intéresser aux vrais codes qui mènent à un déchiffrage non ambigü.

3.2.3 codes

On va définir uniquement des codes binaires c'est-à-dire sur l'alphabet $\{0, 1\}$.

Définition 3.3 *un ensemble de mots C sur l'alphabet $\{0, 1\}$ est un code binaire (à déchiffrage unique) si et seulement si pour toutes suites de mots $u_1 \dots u_p$, $v_1 \dots v_q$ de C on a $u_1 \dots u_p = v_1 \dots v_q \Leftrightarrow p = q$ et $u_i = v_i, i = 1 \dots p$*

***exemple* 3.1** : $C = \{1, 01, 001, 000\}$ est un code mais $C' = \{1, 01, 10, 011\}$ n'est pas un code car on a $(10)(011) = (10)(01)(1)$.

□

Propriété 3.1 *tout ensemble de mots préfixe est un code (un ensemble de mots est préfixe si aucun de ses mots n'est le préfixe (début) d'un autre de ses mots)*

Définition 3.4 *un ensemble de mots de même longueur est un code de longueur fixe.*

remarque : dans un code binaire, le nombre de mots de longueur i est au plus 2^i puisque ces mots appartiennent à $\mathbb{B}^i$.

Propriété 3.2 (inégalité de Kraft) *tout code tel que* $\sum_{c\in C} 2^{-|c|} \leq 1$ *peut être transformé en un code préfixe équivalent (même nombre de mots, même distribution de longueur).*

preuve : La construction du code préfixe équivalent se fera par des choix de chemins dans un arbre binaire dans lequel les branches de gauche seront étiquetées par 0 et celles de droite par 1. Pour assurer un code préfixe, il suffit de choisir des chemins dont aucun n'est inclus dans un autre. Chaque mot du code préfixe correspondra donc à un chemin aboutissant à un noeud de l'arbre.

Dans le code C, il y a des mots de différentes longueur $1, \ldots, p$. Soient $n_1, \ldots n_p$ les nombres de mots du code C de longueur respective $1, \ldots, p$. On a alors

$$\sum_{c\in C} 2^{-|c|} = \sum_{i=1}^{i=p} n_i 2^{-i} \leq 1$$

On en déduit

$n_1 2^{-1} \leq 1$

$n_1 2^{-1} + n_2 2^{-2} \leq 1$

$\ldots$

$n_1 2^{-1} + n_2 2^{-2} + \cdots + n_p 2^{-p} \leq 1$

Donc on a $n_1 \leq 2$, $n_2 \leq 4 - 2n_1$, $\ldots$, $n_p \leq 2^p - (2^{p-1}n_1 + 2^{p-2}n_2 + \cdots + 2^1 n_{p-1})$.

On a alors 3 cas :

- $n_1 = 2$: donc $n_2 = \cdots = n_p = 0$, les deux seuls mots de C sont 0 et 1 qui est un code préfixe
- $n_1 = 1$: C a un mot de longueur 1 et au plus 2 mots de longueur 2, il suffit alors de choisir pour le mot de longueur 1 le mot 0 et il reste deux mots de longueur 2 possibles 10 11
- $n_1 = 0$: C n'a pas de mot de longueur 1 et C a au plus 4 mots de longueur 2, il suffit de choisir parmi 00, 01, 10, 11.

Supposons construits les mots de longueur $1, \ldots, i-1$.

On a donc choisi dans l'arbre des chemins jusqu'au niveau $i-1$.

On sait que $n_i \leq 2^i - (2^{i-1}n_1 + 2^{i-2}n_2 + \cdots + 2^1 n_{i-1})$.

On va montrer que $2^i - (2^{i-1}n_1 + 2^{i-2}n_2 + \cdots + 2^1 n_{i-1})$ est le nombre de nuds du niveau i qui sont sur les chemins poursuivant ceux que l'on a choisi (donc des noeuds descendants de ceux choisis précédemment).

En effet, les descendants des mots de longueur 1 sont au nombre de $2^{i-1}n_1$, les descendants des mots de longueur 2 sont au nombre de $2^{i-2}n_2, \ldots$, les descendants des mots de longueur $i-1$ sont au nombre de $2^{i-(i-1)}n_{i-1} = 2n_{i-1}$.

Il reste donc au plus $2^i - (2^{i-1}n_1 + 2^{i-2}n_2 + \cdots + 2^1 n_{i-1})$ nœuds disponibles. On peut donc choisir n_i mots au niveau i.

Par une récurrence finie on a ainsi construit un code équivalent à C.

■

***exemple* 3.2** : $C = \{10, 11, 000, 101, 111, 1100, 1101\}$. $\sum_{c\in C} 2^{-c} = 2\times 2^{-2}+3\times 2^{-3}+2\times 2^{-4} = 1$.

On utilise un arbre de hauteur 4 (4 est la longueur maximale d'un mot de C). Sur chaque niveau k on choisit autant de chemins qui ne soient pas strictement inclus dans un autre chemin qu'il y a de mots de C de longueur k.

□

Théorème 3.3 (Mac Millan) *tout code vérifie l'inégalité de Kraft.*

preuve : On appelle polynôme caractéristique d'un code $C = \{m_1, \ldots, m_r\}$ le polynôme $P_C(X) = \sum_{j=1,\ldots,r} X^{|m_j|}$. Soit n la longueur d'un plus long mot de C.

Par exemple pour le code $\{101, 01, 11, 00\}$ on a $P = 3X^2 + X^3$.

On va tout d'abord montrer que C^k est aussi un code.

En effet soient $u_1 \ldots u_p, v_1 \ldots v_q \in C^k$ tels que $u_1 \ldots u_p = v_1 \ldots v_q$. Comme chaque mot est dans C^k, on peut écrire $u_i = u_{i,1} \ldots u_{i,k}$ pour $i = 1 \ldots p$ et de même $v_j = v_{j,1} \ldots v_{j,k}$ pour $j = 1 \ldots q$. On en déduit alors

$$u_{1,1} \ldots u_{1,k} \ldots u_{p,1} \ldots u_{p,k} = v_{1,1} \ldots v_{1,k} \ldots v_{q,1} \ldots v_{q,k}$$

égalité dans laquelle chaque mot est dans C.

Puisque C est un code (à déchiffrage unique), on a nécessairement $pk = qk$ donc $p = q$ et $u_{i,l} = v_{i,l}$ pour $i = 1 \ldots p$ et $l = 1 \ldots k$.

Maintenant montrons par récurrence sur k que P_C^k est le polynôme caractéristique de C^k.

C'est vrai pour $k = 1$.

Soit $k \geq 1$, supposons que P_C^k est le polynôme caractéristique de C^k et cherchons le polynôme caractéristique de C^{k+1}.

Chaque mot de C^{k+1} est obtenu par concaténation d'un mot de C^k avec un mot de C. Donc $P_{C^{k+1}}(X) = \sum_{c\in C^{k+1}} X^{|c|} = \sum_{(c,c')\in(C^k\times C)} X^{|c|+|c'|} = \sum_{(c,c')\in(C^k\times C)} X^{|c|}X^{|c'|} = (\sum_{c\in C^k} X^{|c|})(\sum_{c'\in C} X^{|c'|}) = P_{C^k}(X)P_C(X) = P_C^k(X)P_C(X) = P_C^{k+1}(X)$.

Dans P_{C^k}, les degrés varient de k à kn et comme C^k est un code binaire (à déchiffrage unique), P_{C^k} est de la forme $P_{C^k}(X) = \sum_{i=k}^{i=kn} a_i X^i$ avec $a_i \leq 2^i$.

On a donc, pour $X = \frac{1}{2}$, d'une part

$$P_{C^k}(\frac{1}{2}) = \sum_{i=k}^{i=kn} a_i 2^{-i} \leq \sum_{i=k}^{i=kn} 2^i 2^{-i} = \sum_{i=k}^{i=kn} 1 = n$$

et d'autre part

$$P_{C^k}(\frac{1}{2}) = (P_C(\frac{1}{2}))^k = (\sum_{c\in C} 2^{-|c|})^k$$

On en déduit donc que $(\sum_{c\in C} 2^{-|c|})^k \leq n$ pour tout $k \geq 1$, soit $(\sum_{c\in C} 2^{-|c|}) \leq e^{\frac{1}{k}\ln n}$ pour tout $k \geq 1$.

Lorsque k tend vers $+\infty$, $e^{\frac{1}{k}\ln n}$ tend vers 1 donc $(\sum_{c\in C} 2^{-|c|}) \leq 1$. ■

Définition 3.5 *un codage de A par C est une application injective de A dans C.*

Pour une source sans mémoire, coder un mot revient à coder chaque lettre une par une et concaténer le codage. L'efficacité d'un codage sera alors mesurée par $E = \dfrac{H(A)}{m}$ où $m = \sum_{a\in a} p(a)|c(a)|$ est la longueur moyenne des mots du code.

3.3 Premier théorème de Shannon

Théorème 3.4 *Soit A une source discrète sans mémoire d'entropie $H(A)$ codé en binaire par un code de longueur moyenne m. On a alors*

$$H(A) \leq m$$

De plus pour toute source discrète sans mémoire A, il existe un code C codant A de longueur moyenne m tel que $H(A) \leq m < H(A) + 1$.

preuve : soit $c : A \to C$ un codage de A.

$$\begin{aligned}
H(A) - m_C &= -\sum_{a\in A} p(a)\log_2 p(a) - \sum_{a\in a} p(a)|c(a)| \\
&= -\sum_{a\in A} p(a)[\log_2 p(a) + |c(a)|] \\
&= -\sum_{a\in A} p(a)[\log_2 p(a) + \log_2 2^{|c(a)|}] \\
&= -\sum_{a\in A} p(a)\log_2[p(a)2^{|c(a)|}] \\
&= \sum_{a\in A} p(a)\log_2[\frac{1}{p(a)2^{|c(a)|}}]
\end{aligned}$$

or on sait que $\log_2 x \leq \dfrac{x-1}{\ln 2}$ donc on a

$$\begin{aligned}
H(A) - m_C &\leq \frac{1}{\ln 2}\sum_{a\in A} p(a)[\frac{1}{p(a)2^{|c(a)|}} - 1] \\
&\leq \frac{1}{\ln 2}\sum_{a\in A}[2^{-|c(a)|} - p(a)] \\
&\leq \frac{1}{\ln 2}(\sum_{a\in A} 2^{-|c(a)|} - \sum_{a\in A} p(a)) \\
&\leq \frac{1}{\ln 2}(\sum_{a\in A} 2^{-|c(a)|} - 1) \\
&\leq 0
\end{aligned}$$

d'après l'inégalité de Kraft et le théorème de Mac Millan.

Soit A une source discrète sans mémoire. Il faut trouver un code qui vérifie $H(A) \leq m < H(A)+1$. On cherche donc un codage pour chaque lettre a tel que

$$-\sum_{a\in A} p(a)\log_2 p(a) \leq \sum_{a\in a} p(a)|c(a)| \leq -\sum_{a\in A} p(a)\log_2 p(a) + 1$$

On pose $m_a = [-\log_2 p(a)] + 1$ où $[x]$ désigne la partie entière de x.

On aura alors $m_a - 1 \leq -\log_2 p(a) \leq m_a$ et donc $2^{m_a-1} \leq \frac{1}{p(a)} \leq 2^{m_a}$.

On en déduit $2^{-m_a} \leq p(a) \leq 2^{-m_a+1}$ et il vient $\sum_{a\in A} 2^{-m_a} \leq 1$.

On construit alors un arbre binaire dans lequel on choisit un codage de la lettre a sur le niveau m_a. La longueur moyenne de ce code est $m_C = \sum_{a\in A} m_a p(a) \leq \sum_{a\in A} p(a)(-\log_2 p(a)+1) = -\sum_{a\in A} p(a)\log_2 p(a) + 1 = H(A)+1$. ■

***exemple* 3.3** : $A = \{a,b,c,d\}$ avec $p(a) = p(b) = 1/4$, $p(c) = 3/8$ et $p(d) = 1/8$. On calcule $m_a = m_b = 3$, $m_c = 2$ et $m_d = 4$. On obtient par exemple le codage suivant

a	010
b	011
c	00
d	1000

□

Théorème 3.5 *pour toute source discrète sans mémoire il existe un codage dont l'efficacité est arbitrairement proche de 1.*

preuve : l'idée est de coder A^l c'est-à-dire l'alphabet formé de tous les mots composés de lettres de A de longueur l. D'après la propriété précédente il existe un code C_l tel que $H(A^l) \leq m_{C_l} < H(A^l)+1$. On va montrer par récurrence sur l que $H(A^l) = lH(A)$

Lemme 3.1 $H(A^l) = lH(A)$ *pour tout* $l \geq 1$

preuve du lemme : avnt de commencer la récurrence on peut remarquer que, par définition on a $H(A^l) = -\sum_{a_1\ldots a_l\in A} p(a_1\ldots a_l)\log_2 p(a_1\ldots a_l)$ et sachant que la source est sans mémoire il y a indépendance des lettres de A et donc $H(A^l) = -\sum_{a_1\ldots a_l\in A} p(a_1)\ldots p(a_l)\log_2(p(a_1)\ldots p(a_l))$

La propriété est vraie pour $l = 1$.

Supposons que $H(A^{l-1}) = (l-1)H(A)$ pour un entier $l \geq 2$. On a alors

$$\begin{aligned}
H(A^l) &= -\sum_{a_1\ldots a_l\in A} p(a_1)\ldots p(a_l)\log_2(p(a_1)\ldots p(a_l)) \\
&= -\sum_{a_1\ldots a_l\in A} p(a_1)\ldots p(a_l)(\log_2 p(a_1) + \log_2(p(a_2)\ldots p(a_l))) \\
&= \sum_{a_1\in A} p(a_1)\log_2 p(a_1)(-\sum_{a_2\ldots a_l\in A} p(a_2\ldots a_l)) \\
&\quad + \sum_{a_1\in A} p(a_1)(-\sum_{a_2\ldots a_l\in A} p(a_2)\ldots p(a_l)\log_2(p(a_2)\ldots p(a_l)))
\end{aligned}$$

Par hypothèse, on a $-\sum_{a_2 \dots a_l \in A} p(a_2) \dots p(a_l) \log_2(p(a_2) \dots p(a_l)) = H(A^{l-1}) = (l-1)H(A)$ donc il vient

$$\begin{aligned} H(A^l) &= -\sum_{a_1 \in A} p(a_1) \log_2 p(a_1) \Big(\sum_{a_2 \dots a_l \in A} p(a_2 \dots a_l) \Big) + (l-1)H(A) \sum_{a_1 \in A} p(a_1) \\ &= H(A) + (l-1)H(A) = lH(A) \end{aligned}$$

■

On a alors pour le code C_l, $H(A) \leq \frac{m_{C_l}}{l} \leq H(A) + \frac{1}{l}$. Le nombre $m = \frac{m_{C_l}}{l}$ représente alors le nombre moyen de symboles binaires pour le codage d'une lettre puisque m_{C_l} représente le nombre moyen de symboles binaires pour le codage de l lettres. Et on a $1 - \frac{1}{l}\frac{H(A)}{m} \leq 1$. ■

On va maintenant donner quelques exemples de codes dont certains ne sont pas binaires.

3.4 code à longueur fixe

Soit une source A de cardinal n. Pour coder A sur $\{0,1\}$ avec un nombre r fixe de symboles binaires il faut avoir que 2^r soit supérieur au cardinal de A. Donc on doit avoir $r \geq \log_2 n$. Pour optimiser son efficacité on choisit alors r tel $\log_2 n \leq r \leq \log_2 n + 1$ et l'efficacité d'un tel code est alors inférieure à $\dfrac{H(A)}{\log_2 n}$. On aura égalité si $n = 2^r$ et si la loi sur A est uniforme.

***exemple* 3.4** : $A = \{a, b, c, d, e\}$ avec une loi uniforme. Il faut $[\log_2 5] + 1 = 3$ bits pour coder A : par exemple

a	010
b	011
c	000
d	100
e	101

□

L'efficacité est alors égale à $E = \frac{\log_2 5}{3} \approx 0,77$.

On peut améliorer cette efficacité si on choisit comme source A^2 avec toujours une loi uniforme : on a alors 25 lettres et donc $r > 2\log_2 5$, soit $r = 5$. L'efficacité devient alors $E = \frac{2\log_2 5}{5} \approx 0,93$.

3.5 code Morse

C'est un code ternaire : les trois symboles utilisés sont le point, le trait et la pause (permettant de séparer les lettres), chacun correspondant à une impulsion électrique de durée croissante. Chaque lettre est codéeselon sa fréquence d'utilisation en anglais. Par exemple, A est codé par .– et E par ..

3.6 code arithmétique

Le message complet est codé par un nombre décimal en virgule flottante.

Par exemple, on doit coder MATT DAMON.

On va estimer les probabilités des lettres source en les assimilant à leur fréquence d'apparition dans le message. On a alors

A	D	M	N	O	T	espace
2/10	1/10	2/10	1/10	1/10	2/10	1/10

On attribue ensuite à chaque lettre un intervalle contenu dans $[0, 1]$: les intervalles doivent être fermés à gauche, ouverts à droite, disjoints 2 à 2 et leurs longueurs doivent correspondre à la fréquence d'apparition de la lettre associée.

On obtient

A	D	M	N	O	T	espace
[0,1 ;0,3[	[0,3 ;0,4[	[0,4 ;0,6[	[0,6 ;0,7[	[0,7 ;0,8[	[0,8 ;1[	[0 ;0,1[

Le nombre qui représentera le message va appartenir à l'intervalle correspondant à la première lettre soit $I_1 = [0,4; 0,6[$. Pour affiner la valeur cherchée, cet intervalle sera restreint par la deuxième lettre, ici A, de la façon suivante :

- on ajoute à la borne inférieure de I_1 la borne inférieure de l'intervalle de la deuxième lettre multipliée par la longueur de I_1.
- on ajoute à la borne inférieure de I_1 la borne supérieure de l'intervalle de la deuxième lettre multipliée par la longueur de I_1.

On obtient ainsi un nouvel intervalle que l'on notera I_2. On applique à I_2 la même restriction avec la troisième lettre et ainsi de suite jusqu'à la dernière restriction.

Cela donne

- pour M, $I_1 = [0,4; 0,6[$
- pour MA, $I_2 = [0,4 + 0,1 \times 0,2; 0,4 + 0,3 \times 0,2[= [0,42; 0,46[$
- pour MAT, $I_3 = [0,42 + 0,8 \times 0,04; 0,42 + 1 \times 0,04[= [0,452; 0,46[$
- pour MATT, $I_4 = [0,452 + 0,8 \times 0,008; 0,452 + 1 \times 0,008[= [0,4584; 0,46[$
- pour MATT , $I_5 = [0,4584 + 0,8 \times 0,0016; 0,4584 + 1 \times 0,0016[= [0,45968; 0,46[$
- pour MATT D, $I_6 = [0,45968+0,3\times0,00032; 0,45968+0,4\times0,00032[= [0,459776; 0,459808[$
- pour MATT DA, $I_6 = [0,459776+0,1\times0,000032; 0,459776+0,3\times0,000032[= [0,4597792; 0,4597856[$
- pour MATT DAM, $I_6 = [0,4597792 + 0,4 \times 0,0000064; 0,4597792 + 0,6 \times 0,0000064[= [0,45978176; 0,45978304[$
- pour MATT DAMO, $I_6 = [0,45978176+0,7\times0,00000128; 0,45978176+0,8\times0,00000128[= [0,459782656; 0,459782784[$
- pour MATT DAMON, $I_6 = [0,459782656 + 0,6 \times 0,000000128; 0,459782656 + 0,7 \times 0,000000128[= [0,4597827328; 0,4597827456[$

La borne inférieure du dernier intervalle est choisie pour coder le message, soit $x = 0,4597827328$.

Pour décoder le message, on commence par déterminer dans quel intervalle se trouve le nombre reçu. Il appartient à $[0,4; 0,6[$ donc la première lettre est M. Ensuite on modifie x en lui soustrayant la borne inférieure de la première lettre, soit $0,4$ et on divise le résultat par la

longueur de l'intervalle de M : on obtient $x = (0,4597827328 - 0,4) \div 2 = 0,29891164$. La deuxième lettre correspond à l'intervalle auquel appartient x, soit l'intervalle $[0,1;0,3[$: c'est donc A. Ainsi de proche en proche on décode le message.

Ici, on a choisi des intervalles déterminés par le message lui-même mais en pratique les intervalles sont fixés au préalable afin d'être connu du récepteur.

3.7 code de Shannon Fano

Cet algorithme construit un code d'une source discrète sans mémoire A en suivant les étapes suivantes :

1. on classe les lettres de A par ordre décroissant de probabilité
2. on forme 2 classes dans A de façon à ce que les probabilités de chaque classe soient les plus proches possibles puis on attribue 1 à la classe du haut et 0 à celle du bas
3. **si** chaque classe a 1 élément **alors** on arrête **sinon** on applique l'étape 2 à chaque ayant plus d'un élément

***exemple* 3.5** : $A = \{a, \ldots, g\}$ avec une loi $\{0,4; 0,2; 0,15; 0,1; 0,05; 0,05; 0,05\}$.

a	0,4	1	1			11
b	0,2	1	0			10
c	0,15	0	1	1		011
d	0,1	0	1	0		010
e	0,05	0	0	1	1	0011
f	0,05	0	0	1	0	0010
g	0,05	0	0	0	0	0000

□

3.8 code de Huffman

Cet algorithme construit un code d'une source discrète sans mémoire A en suivant les étapes suivantes :

1. on classe les lettres de A par ordre croissant de probabilité
2. on relie 2 lettres de plus faible probabilité par 2 arêtes à un noeud parent qui remplace ces 2 lettres et est affecté de la somme des probabilités
3. **si** on recommence l'étape 2 jusqu'à arriver à la racine de l'arbre

***exemple* 3.6** : $A = \{a, \ldots, g\}$ avec une loi $\{0,4; 0,2; 0,15; 0,1; 0,05; 0,05; 0,05\}$.

g	f	e	d	c	b	a
0,05	0,05	0,05	0,1	0,15	0,2	0,4
0,1		0,05	0,1	0,15	0,2	0,4
0,15			0,1	0,15	0,2	0,4
0,25				0,15	0,2	0,4
0,25				0,35		0,4
0,6						0,4
1						

Le codage est alors le suivant

g	f	e	d	c	b	a
00000	00001	0001	001	010	011	1

avec une longueur moyenne de $2,45$

□

Chapitre 4

Canaux

On se limitera à la modélisation des canaux discrets.

4.1 Définition

De façon générale pour définir un canal de transmission on a besoin des entrées, des sorties et du bruit qui peut perturber les transmissions du canal.

Pour les canaux discrets, les entrées et les sorties seront modélisées par des alphabets finis A et B respectivement.

Quant au bruit il sera modélisé par une probabilité conditionnelle de B sachant A. En effet, la sortie doit idéalement être déterminée par l'entrée mais plus il y a de perturbation moins la sortie dépend de l'entrée.

On dira qu'un canal discret est sans mémoire si le bruit est indépendant du temps.

On définit alors un canal discret sans mémoire par la donnée de
- A un alphabet d'entrée
- B un alphabet de sortie
- M une matrice telle que $M_{i,j} = p(b_j/a_i)$

remarque si B est indépendant de A ce qui signifie que le signal reçu est totalement aléatoire par rapport au signal envoyé alors on dira que le canal est inutile ; cela se traduit par

$$I(A = a_i, B = b_j) = \log_2 \frac{p(b_j/a_i)}{p(b_j)} = \log_2 1 = 0$$

.

On peut alors préciser plusieurs caractéristiques. Un canal est
- sans perte si l'entrée est entièrement déterminée par la sortie
- déterministe si la sortie est entièrement déterminée par l'entrée
- sans bruit si il est sans perte et déterministe
- symétrique si chaque ligne (colonne) contient les mêmes valeurs à une permutation près
- inutile si toutes les lignes sont identiques

***exemple* 4.1** : $M = \begin{pmatrix} 0,1 & 0,9 \\ 0,9 & 0,1 \\ 0,1 & 0,9 \end{pmatrix}$ est la matrice d'un canal symétrique.

□

4.2 Capacité d'un canal

4.2.1 Définition

On veut modéliser l'idée suivante : quelle quantité d'information maximale peut transiter par un canal par unité de temps ?

Autrement dit si A est la source et B ce qui sort du canal, quelle quantité d'information peut-on obtenir au maximum sur A connaissant B ?

Si on considère A, B comme deux variables aléatoires à valeurs respectives dans $\{a_1, \ldots, a_n\}$ et $\{b_1, \ldots, b_p\}$ alors on peut définir l'information mutuelle jointe comme l'espérance de l'information mutuelle $I(A = a_i, B = b_j)$ soit $I(A; B) = \sum_{i,j} p(a_i, b_j) I(A = a_i, B = b_j)$

On a les formulations équivalentes suivantes

$$I(A;B) = \sum_{i,j} p(a_i,b_j) \log_2 \frac{p(a_i/b_j)}{p(a_i)} = \sum_{i,j} p(a_i,b_j) \log_2 \frac{p(b_j/a_i)}{p(b_j)} = \sum_{i,j} p(a_i,b_j) \log_2 \frac{p(a_i,b_j)}{p(a_i)p(b_j)}$$

Remarque : $p(b_j/a_i)$ est donné par $M_{i,j}$ et $p(b_j) = \sum_i p(a_i, b_j) = \sum_i p(a_i) p(b_j/a_i)$ donc la loi de B est déterminée par le canal et la loi de A. En conséquence, l'information mutuelle jointe ne dépend que du canal et de la loi de A.

On a les propriétés suivantes :

Propriété 4.1 $I(A; B) \geq 0$

preuve :

$$\begin{aligned}
-I(A;B) &= -\sum_i \sum_j p(a_i,b_j) \log_2 \frac{p(a_i,b_j)}{p(b_j)p(a_i)} \\
&= \sum_i \sum_j p(a_i,b_j) \log_2 \frac{p(b_j)p(a_i)}{p(a_i,b_j)} \\
&= \frac{1}{\ln 2} \sum_i \sum_j p(a_i,b_j) \ln \frac{p(b_j)p(a_i)}{p(a_i,b_j)} \\
&\leq \frac{1}{\ln 2} \sum_i \sum_j p(a_i,b_j) [\frac{p(b_j)p(a_i)}{p(a_i,b_j)} - 1] \\
&\leq \frac{1}{\ln 2} [\sum_i \sum_j p(a_i,b_j) \frac{p(b_j)p(a_i)}{p(a_i,b_j)} - \sum_i \sum_j p(a_i,b_j)] \\
&\leq \frac{1}{\ln 2} [\sum_i \sum_j p(b_j)p(a_i) - 1] = 0
\end{aligned}$$

■

Propriété 4.2 $I(A; B) = H(B) - H(B/A) = H(A) - H(A/B) = H(A) + H(B) - H(A, B)$

preuve :

$$\begin{aligned} H(A) - H(A/B) &= -\sum_i p(a_i)\log_2 p(a_i) - \sum_j p(b_j)H(A/b_j) \\ &= -\sum_i p(a_i)\log_2 p(a_i) \\ &\quad -\sum_j p(b_j)(-\sum_i p(a_i/b_j)\log_2 p(a_i/b_j)) \\ &= \sum_i[-p(a_i)\log_2 p(a_i) + \sum_j p(b_j)p(a_i/b_j)\log_2 p(a_i/b_j)] \end{aligned}$$

Or $p(a_i/b_j) = \dfrac{p(a_i,b_j)}{p(b_j)}$ donc on a

$$H(A) - H(A/B) = \sum_i[-p(a_i)\log_2 p(a_i) + \sum_j p(a_i,b_j)\log_2 p(a_i,b_j) - \sum_j p(a_i,b_j)\log_2 p(b_j)]$$

Mais $-\sum_i p(a_i)\log_2 p(a_i) = -\sum_i\sum_j p(a_i,b_j)\log_2 p(a_i)$, d'où

$$\begin{aligned} H(A) - H(A/B) &= \sum_i\sum_j[-p(a_i,b_j)\log_2 p(a_i) + p(a_i,b_j)\log_2 p(a_i,b_j) - p(a_i,b_j)\log_2 p(b_j) \\ &= \sum_i\sum_j p(a_i,b_j)\log_2\frac{p(a_i,b_j)}{p(b_j)p(a_i)} \\ &= I(A;B) \end{aligned}$$

La dernière égalité est une conséquence de la propriété 2.6. ■

Conséquence : Le bruit du canal sera sans influence si connaissant B on peut retrouver A sans ambiguïté donc si $H(A/B) = 0$ ($H(A/B)$ mesurant l'incertitude sur A connaissant B). Donc on aura dans ce cas $I(A;B) = H(A) - H(A/B) = H(A)$ maximal.

On considère alors toutes les sources possibles et l'information mutuelle jointe pour chacune de ces sources : la capacité sera alors le maximum de ces valeurs.

Prendre toutes les sources possibles A c'est faire varier la loi de probabilité p_A.

Définition 4.1 $C = max_{p_A} I(A;B)$

C est un réel positif ou nul inférieur à $H(A)$ et à $H(B)$ donc à $\log_2|A|$ et à $\log_2|B|$ d'après la propriété 2.4.

On va voir que pour un canal symétrique le calcul de la capacité peut être réalisé.

4.2.2 Canal symétrique

Propriété 4.3 *Pour un canal symétrique, $H(B/A)$ ne dépend pas de la distribution p_A.*

preuve :

$$\begin{aligned}
H(B/A) &= -\sum_{a\in A}\sum_{b\in B} p(a,b)\log_2 p(b/a) \\
&= -\sum_{a\in A}\sum_{b\in B} p(a)p(b/a)\log_2 p(b/a) \\
&= -\sum_{a\in A} p(a)[\sum_{b\in B} p(b/a)\log_2 p(b/a)] \\
&= -\sum_{a\in A} p(a)[\sum_{j=1\ldots|B|} p_j \log_2 p_j] \\
&= -\sum_{j=1\ldots|B|} p_j \log_2 p_j
\end{aligned}$$

■

Pour maximiser $I(A;B) = H(B) - H(B/A)$, il suffit alors de maximiser $H(B)$. Or on sait que $H(B) \leq \log_2 |B|$ et qu'il y a égalité *si et seulement si* la loi est uniforme sur B. On cherchera donc une loi sur A telle que la loi sur B soit uniforme connaissant M.

Supposons que p_A soit uniforme. On a alors $p(b) = \sum_a p(a,b) = \sum_a p(b/a)p(a) = \frac{1}{|A|}\sum_a p(b/a)$.

Or $\sum_a p(b/a)$ est la somme des éléments de M sur la colonne d'indice b et cette quantité est la même pour toutes les colonnes puisque le canal est symétrique. Donc la loi sur B est alors uniforme et de plus $H(B) = \log_2 |B|$.

Donc la capacité d'un canal symétrique est

$C_{sym} = \log_2 |B| + \sum_{j=1\ldots|B|} p_j \log_2 p_j$

***exemple* 4.2** : Soit $B = A = \{0,1\}$ et $M = \begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix}$

Cela signifie que $B = A$ avec une probabilité $1-p$ (donc p représente la probabilité de transmettre une erreur).

$C_{sym} = \log_2 2 + p\log_2 p + (1-p)\log_2(1-p) = 1 + p\log_2 p + (1-p)\log_2(1-p)$

□

4.2.3 Exemple d'un canal avec bruit

Soit $A = \{a,b,c,d\}$ et $B = \{0,1\}$ avec

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

$I(A;B) = H(B) - H(B/A)$ or $H(B/A) = 0$ car il n'y a pas d'incertitude sur B sachant A.

En effet, par définition on a

$H(B/A) = p(a)H(B/A = a) + p(b)H(B/A = b) + p(c)H(B/A = c) + p(d)H(B/A = d)$,

soit

$H(B/A) = p(a)[-p(B = 0/A = a)\log_2 p(B = 0/A = a) - p(B = 1/A = a)\log_2 p(B = 1/A = a)] + p(b)[-p(B = 0/A = b)\log_2 p(B = 0/A = b) - p(B = 1/A = b)\log_2 p(B = 1/A = b)] + p(c)[-p(B = 0/A = c)\log_2 p(B = 0/A = c) - p(B = 1/A = c)\log_2 p(B = 1/A = c)] + p(d)[-p(B = 0/A = d)\log_2 p(B = 0/A = d) - p(B = 1/A = d)\log_2 p(B = 1/A = d)]$.

Avec les valeurs de M, cela donne

$H(B/A) = p(a)[-1\log_2 1 - 0\log_2 0] + p(b)[-1\log_2 1 - 0\log_2 0] + p(c)[-0\log_2 0 - 1\log_2 1] + p(d)[-0\log_2 0 - 1\log_2 1] = 0$.

Il reste donc $I(A;B) = H(B) = -p(0)\log_2 p(0) - p(1)\log_2 p(1)$

La loi de B se calcule ainsi

$p(0) = p(B = 0, A = a) + p(B = 0, A = b) + p(B = 0, A = c) + p(B = 0, A = d) = p(a)p(B = 0/A = a) + p(b)p(B = 0/A = b) + p(c)p(B = 0/A = c) + p(d)p(B = 0/A = d) = p(a) + p(b)$.

De même, $p(1) = p(c) + p(d) = 1 - (p(a) + p(b))$.

D'où $I(A;B) = -(p(a) + p(b))\log_2(p(a) + p(b)) - (1 - (p(a) + p(b)))\log_2(1 - (p(a) + p(b)))$.

La fonction $x \mapsto x\ln x + (1 - x)\ln(1 - x)$ sur l'intervalle $]0, 1[$ atteint son maximum pour $x = 0,5$.

Donc $I(A;B)$ vaut au maximum 1.

4.3 Deuxième théorème de Shannon

On va donner une version sans preuve de ce résultat

Soit un canal de capacité C qui transmet d_C symboles par unité de temps d'une source A qui elle a un débit de d_S symboles par unité de temps.

Il existe une méthode de codage de A qui garantit une probabilité d'erreur de décodage arbitrairement faible si $d_S H(A) < d_C C$.

La preuve de ce théorème ne fournit pas de méthode.

***exemple* 4.3** : soit $A = \{0, 1\}$ avec $p(0) = 0,98$ et $p(1) = 0,02$ et un débit de $600kbits/s$. On a alors $H(A) = -0,98\log_2 0,98 - 0,02\log_2 0,02 = 0,1414$ et $d_S H(A) = 84840bits/s$

Soit un canal binaire symétrique avec $p = 0,001$ de débit $450kbits/s$. On a alors $C = 1 + p\log_2 p + (1 - p)\log_2(1 - p) = 0,9886$ et $d_C C = 444870bits/s$.

La condition $d_S H(A) < d_C C$ est vérifiée.

□

4.4 Pourquoi doit-on coder avant le canal

***exemple* 4.4** : soit $A = B = \{0, 1\}$ avec $p(A = 1) = q$ et $p(A = 0) = 1 - q$ et un canal binaire symétrique avec $M = \begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix}$

On va comparer une transmission sans codage préalable avant passage par le canal avec une transmission précédée d'un code à répétition que l'on précisera.

Transmission sans codage :

la probabilité d'obtenir une erreur est alors

$P_e = p(B = 0, A = 1) + p(B = 1, A = 0) = p(A = 1)p(B = 0/A = 1) + p(A = 0)p(B = 1/A = 0) = (1 - q)p + qp = p$

Transmission avec codage à répétition :

0 est codée par 000 et 1 est codé par 111 (chaque bit est répété 2 fois).

A la sortie du canal on considère donc des mots de 3 bits qu'il faut décoder pour retrouver le bit initialement émis. Mais ces mots de trois bits peuvent être quelconques ($\{000, 001, 010, 011, 100, \cdots\}$).

Le décodage consiste à choisir pour symbole émis l'élément binaire qui figure au moins 2 fois dans le mot reçu (mot constitué de 3 symboles binaires).

Soient A' B' les variables aléatoires correspondant respectivement aux mots d'entrée et de sortie.

La probabilité d'obtenir une erreur est alors

$$\begin{aligned}
P_e &= p(B' = 000, A' = 111) + p(B' = 100, A' = 111) + p(B' = 010, A' = 111) + p(B' = 001, A' = 111) + \\
&\quad p(B' = 111, A' = 000) + p(B' = 011, A' = 000) + p(B' = 101, A' = 000) + p(B' = 110, A' = 000) \\
&= p(B' = 000/A' = 111)p(A' = 111) + p(B' = 100/A' = 111)p(A' = 111) + \\
&\quad p(B' = 010/A' = 111)p(A' = 111) + p(B' = 001/A' = 111)p(A' = 111) + \\
&\quad p(B' = 111/A' = 000)p(A' = 000) + p(B' = 011/A' = 000)p(A' = 000) + \\
&\quad p(B' = 101/A' = 000)p(A' = 000) + p(B' = 110/A' = 000)p(A' = 000) \\
&= q(p^3 + 3p^2(1 - p)) + (1 - q)((p^3 + 3p^2(1 - p))) \\
&= -2p^3 + 3p^2
\end{aligned}$$

Si on compare les 2 valeurs p et $-2p^3 + 3p^2$, on constate que
- si $p < 0,5$ alors $-2p^3 + 3p^2 < p$ donc on obtient une probabilité d'erreur plus faible avec le codage
- si $p = 0,5$ alors les 2 transmissions sont équivalentes
- si $p > 0,5$ alors la transmission sans codage est alors préférable

Mais on peut remarquer que dans le dernier cas il suffit de permuter les lettres de l'alphabet de sortie ce qui changera p en $1 - p$.

□

Les avantages du code ici sont de deux ordres

– détection d'erreur : si on reçoit un mot qui contient à la fois 0 et 1 alors on est sûr qu'il y a eu une erreur de transmission puisqu'on ne devrait recevoir que 000 ou 111. De plus, on peut remarquer que l'on détectera 1 ou 2 erreurs mais pas 3 car alors on recevrait un mot possiblement envoyé (000 au lieu de 111 par exemple)
– correction d'erreur : dans le cas où une et une seule erreur est commise alors on pourra corriger cette erreur en examinant les 2 bits majoritaires. Mais ce n'est pas possible si 2 erreurs ont été commises

On dira que le code à répétition est 1-correcteur et 2-détecteur.

De façon générale pour mesurer un codage de canal on utilisera la notion suivante

Définition 4.2 *Le rendement d'un code de canal calcule le rapport entre le nombre de bits nécessaires pour coder N symboles et le nombre de bits n réellement utilisés*

$$R = \frac{\log_2 N}{n}$$

Par exemple si on a $N = 2^m = 2^{10}$ et $n = 10$ alors $R = 1$ mais si $n = 20$ (soit $k = m$) alors $R = 1/2$ c'est-à-dire que le codage de canal double le nombre de bits des symboles de l'entrée.

Pour le code à répétition ci-dessus $\log_2 N = 1$ et $n = 3$ donc $R = 1/3$.

4.5 Codages détecteurs

Le plus simple des codes détecteur utilise un bit de parité : on consière que l'on transmet des blocs de $p - 1$ bits et on les code en rajoutant à la fin un p-ième bit choisi de façon que le nombre total de bits ègaux à 1 dans le mot de p bits transmis soit pair.

A la reception, si le nombre de bits dans un mot de p bits est impair on aura alors détecter au moins une erreur.

4.6 Codages correcteurs

On s'intéressera au codage par blocs : chaque mot de longueur m (éléments binaires) sera codé par un mot de longueur fixe $n \geq m$.

On généralise l'exemple 4.4 en distinguant dans un mot code les éléments binaires d'information et les éléments binaires de contrôle calculés à partir des éléments binaires d'information.

On a donc en source des mots de longueur m qui seront chacun codé par un code à longueur fixe n dans lesquels figurent m bits d'information et k bits de contrôle ($n = m + k$).

On aura donc le schéma de transmission suivant

$$i_1 i_2 \dots i_m \underbrace{\rightsquigarrow}_{codage} y_1 y_2 \dots y_n \dashrightarrow canal \dashrightarrow y'_1 y'_2 \dots y'_n \underbrace{\rightsquigarrow}_{decodage} i'_1 i'_2 \dots i'_m$$

Parmi les 2^n mots $y'_1 y'_2 \dots y'_n$ reçus, il n'y en a que 2^m qui sont corrects il y a donc potentiellement $2^n - 2^m$ mots erronés.

On va détailler le code de Hamming[7,4] ($m = 4$, $k = 3$, $n = 7$) dont le rendement est $4/7$ et qui est 1-correcteur.

On considère la matrice $H = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}$

qui a été obtenue par un calcul polynomial sur le corps $\mathbb{Z}/2\mathbb{Z}$.

codage

- chaque mot de 4 bits $i_1i_2i_3i_4$ est tout d'abord complété par trois 0 en préfixe et on obtient un vecteur a de 7 bits $000i_1i_2i_3i_4$
- on calcule le vecteur $r = Ha$
- le mot code est alors $c = ri_1i_2i_3i_4$

***exemple* 4.5** : soit à coder 1010.

- $a = 0001010$
- $r = Ha = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$
- le mot code est $c = 0011010$

□

décodage : soit t le mot de 7 bits reçu

- on calcule $s = Ht$
- si $s = 0$ alors on garde les 4 derniers bits de t comme résultat du décodage de la transmission
- sinon s correspond à une colonne j de H et on change alors le bit j de t et on en garde les 4 derniers bits comme résultat du décodage de la transmission.

***exemple (suite)* 4.6** : soit $t = 0010010$ le mot reçu.

- $s = Ht = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$
- s correspond à la colonne 4 de H, on change le bit 4 de t donc le mot décodé est 1010 ce qui est correct malgré l'erreur sur ce bit.

□

Ce code de Hamming ne fonctionne que si une erreur au plus s'est produite.

Pour le cas général, une matrice $k \times n$ est utilisée sur le même principe.

Il est appliqué au Minitel avec $m = 120$, $k = 7$ en rajoutant un bit de parité et un octet formé de 8 zéros qui permet de détecter les pannes techniques importantes (foudre ...). Son rendement est 120/136.