Théorie de l'information
Ensimag 1A

Contents

Part I

Cours de théorie de l'information

Introduction

Initiée par Claude Shannon en 1948, la théorie de l'information est une modélisation mathématique, essentiellement probabiliste, des problèmes liés à la mesure de la quantité d'information contenue dans un message, au stockage efficace et fiable d'un message (compression et protection).

Depuis son origine, la théorie de l'information est liée à celle de la communication. Quoique ce champ applicatif soit toujours parfaitement actuel, les outils et méthodes de la théorie de l'information trouvent leur utilité dans de nombreux autres domaines tels que l'estimation ou la finance.

Objectif du cours

Fournir les notions théoriques de base pour la mesure quantitative de l'information (qu'est-ce qu'un bit d'information ?), sa représentation, son stockage, sa transmission, sa protection et sa dissimulation. Application à la compression sans perte, au codage correcteur, à la sécurité des transmissions et des contenus (stéganographie).

Le cours aborde les aspects suivants :

- Mesurer l'information. Incertitude et information. Entropies. L'entropie de Shannon et ses propriétés. Entropie de lois composées et transfert d'information.
- Structure d'une chaîne de communication. Sources d'information et compression. Canaux, capacité et codage de canal.
- Codage des sources discrètes. Equipartition asymptotique, notion de suite typique. Codes optimaux théoriques. Construction effective de codes optimaux.
- Transmettre et stocker l'information. Canal discret (canal binaire symétrique). Codage de canal et second théorème de Shannon.
- Codes détecteurs et correcteurs d'erreurs. Répétition et second théorème de Shannon. Codes détecteurs d'erreur. Codes correcteurs d'erreur. Codes en blocs linéaires. Distance de Hamming et distance Euclidienne. Décodage au sens du maximum de vraisemblance. Codes convolutifs et algorithme de Viterbi. Diagramme d'état, treillis et algorithme de Viterbi.
- Numérisation : échantillonnage et quantification.

Des domaines d'application seront évoqués pour illustrer l'intérêt des concepts abordés, citons :

- Stéganographie, tatouage, fuite d'information et sécurité.
- Compression d'un message. L'exploitation des propriétés statistiques d'un message peut permettre sa compression. Celle-ci s'avère utile dans de nombreux cas : transmission sur un canal à débit limité, stockage sur un support de capacité limitée ...
- Codes correcteurs d'erreurs. L'ajout d'une certaine redondance dans un message permet au lecteur de déceler un certain nombre d'anomalies (détection d'erreur) et parfois même de les corriger. Les codes correcteurs sont couramment utilisés : lecteurs de disques compacts par exemple.

- Capacité. Exploitation au mieux des caractéristiques d'un canal de transmission. La théorie de l'information fournie des bornes supérieures pour le débit d'information maximum qu'il est possible de faire passer au travers d'un canal physique donné.

Chapter 1

Incertitude, Entropie et Information

Contents

1.1 Notations

Au sens de Shannon, la théorie de l'information repose de façon essentielle sur l'existence d'une mesure objective de la quantité d'information contenue dans un message aléatoire. Bien que cette approche du problème de la mesure quantitative de l'information n'englobe pas tous les aspects du problème, elle satisfait à certaines attentes intuitives.

On considère une variable aléatoire discrète X prenant ses valeurs dans l'alphabet $\mathcal{A}$.
Pour tout $x \in \mathcal{A}$, on note $p(x) = p_X(x) = Pr[X = x]$ la probabilité de l'éventualité x.
On note les états $x_1, \cdots, x_N$ et leurs probabilités $Pr[X = x_j] = p(x_j) = p_j$.
$\mathcal{P} = \{p(x)\}_{x\in\mathcal{A}}$ est la loi de la v.a. X.
Les $\{p(x)\}_{x\in\mathcal{A}}$ sont des probabilités *a priori* sur les réalisations possibles de la v.a. X.

1.2 Des variables plus aléatoires que d'autres.

Dans une épreuve aléatoire, chaque éventualité se produit avec une fréquence connue *a priori*. Si l'on cherche à deviner l'issue de l'épreuve, les chances de succès dans cette tentative varient selon la distribution de probabilité sur l'ensemble des états possibles. La meilleure prédiction du résultat d'un tirage, au sens du minimum de la probabilité d'erreur, est l'état le plus probable. Si la probabilité de cet état est $p_{\max}$, la probabilité d'erreur est $1 - p_{\max}$.

Prenons l'exemple d'une loi de Bernoulli avec $p_1 = p$ et $p_2 = 1 - p$. Pour $p = 1$ on prédit l'état 1 et la probabilité d'erreur est nulle. Il en est de même pour $p = 0$: on prédit l'état 2 et la probabilité d'erreur est nulle. Entre ces deux extrêmes pour lesquels le résultat de l'expérience aléatoire est certain, le résultat est d'autant plus incertain (la probabilité d'erreur d'autant plus grande) que p est proche de $1/2$.

1.3 Définition de l'entropie de Shannon.

Incertitude de chacun des états. Il est assez naturel de définir l'incertitude $i(x)$ liée à la réalisation de l'état x comme étant une fonction de sa probabilité *a priori*.

$$i(x) = F\left[p(x)\right]$$

On peut raisonnablement imposer à F les propriétés suivantes :

- $i(x)$ est une quantité positive.
- F est une fonction décroissante, c'est-à-dire que l'incertitude est d'autant plus élevée que la probabilité *a priori* d'apparition de x est faible.
- Si X et Y sont deux v.a. indépendantes, l'incertitude liée à la réalisation du couple d'états (x, y) est la somme de l'incertitude liée à x et de celle liée à y :

$$F\left[p(x,y)\right] = F\left[p(x)p(y)\right] = F\left[p(x)\right] + F\left[p(y)\right]$$

L'utilisation de ces conditions conduit facilement à $F(x) = -\alpha \log(x)$ avec $\alpha > 0$.

Incertitude moyenne de l'ensemble des états. Pour définir une incertitude de la v.a. elle-même, il est naturel d'évaluer la moyenne des incertitudes sur l'ensemble des états. On définit de cette façon une quantité, l'entropie, attachée à l'expérience aléatoire elle-même. L'entropie est associée à la v.a. X, ou de façon équivalente à sa loi $\mathcal{P}$, les notations $H(X)$ ou $H(\mathcal{P})$ sont employées :

$$H\left(X\right) = H(\mathcal{P}) = -\alpha \sum_{x \in \mathcal{A}} p(x) \log p(x)$$

On adopte la convention $0 \log_2 0 = 0$ (prolongement par continuité de $x \log_2 x$ en 0).

Exercice : construction axiomatique de l'entropie. Montrer qu'une suite de fonctions $H_n(p_1, \cdots, p_n)$ symétriques qui vérifient les 3 propriétés suivantes :

1. $H_2(\frac{1}{2}, \frac{1}{2}) = 1$,
2. $H_2(p, 1-p)$ est une fonction continue de p,
3. $H_n(p_1, \cdots, p_n) = H_{n-1}(p_1 + p_2, p_3, \cdots, p_n) + (p_1 + p_2) H_2\left(\frac{p_1}{p_1+p_2}, \frac{p_2}{p_1+p_2}\right)$

est de la forme $H_n(p_1, \cdots, p_n) = -\sum_{j=1}^{n} p_j \log_2 p_j$

Cas de la loi de Bernoulli. L'expérience aléatoire la plus simple qui soit comporte deux issues possibles (pour une seule issue, l'expérience n'est pas aléatoire), c'est la loi de Bernoulli.

Les deux issues possibles $x_1 = 0$ et $x_2 = 1$, apparaissent avec probabilités $p(x_1) = p$ et $p\left(x_2\right) = 1 - p$. L'entropie s'écrit $H\left(X\right) = \alpha\left[-p \log\left(p\right) - (1-p) \log\left(1-p\right)\right]$. Cette entropie est maximale lorsque les deux issues de l'expérience sont équiprobables $p = 1 - p = 1/2$. Choisir α, c'est choisir l'unité de mesure de l'incertitude. Le choix le plus couramment adopté attribue une incertitude de 1 bit à l'expérience aléatoire la plus simple qui soit : le pile ou face équitable. Avec ce choix du bit en tant qu'unité de mesure, l'entropie de la loi de Bernoulli prend la forme :

$$H\left(X\right) = -p \log_2\left(p\right) - (1-p) \log_2\left(1-p\right) \text{ bit(s)}$$

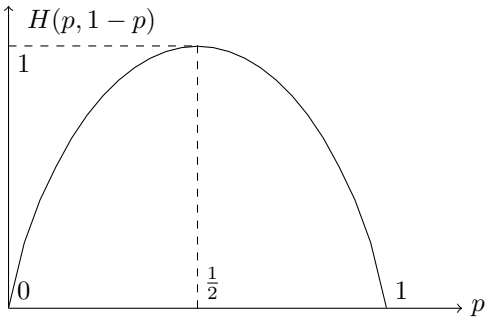


Figure 1.1: Entropie d'une loi de Bernoulli.

La fonction

$$H(X) = -p\log_2(p) - (1-p)\log_2(1-p) \text{ bit(s)}$$

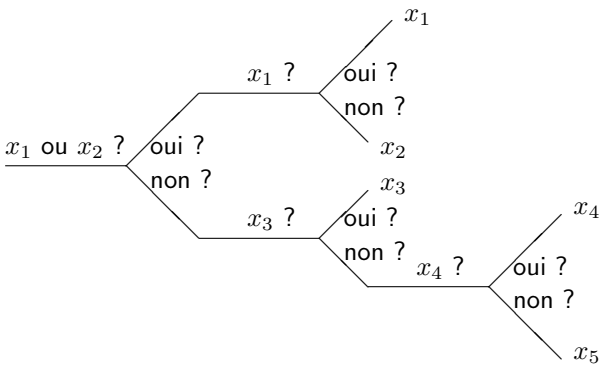
est

- continue en la variable p,
- nulle en $p = 0$ et $p = 1$ (loi déterministe),
- maximale en $p = 1/2$ (loi uniforme),
- strictement concave sur l'intervalle $[0, 1]$.

1.4 Applications à la construction d'un questionnaire

Exemple de bon questionnaire sous optimal

Considérons une v.a. X pouvant prendre cinq états $x_i, i = 1 \cdots 5$ avec les probabilités $p_1 = 0.3, p_2 = 0.2, p_3 = 0.2, p_4 = 0.15, p_5 = 0.15$. L'expérience X étant réalisée, on cherche à en déterminer le résultat à l'aide de questions binaires (à deux réponses). Pour cela, on construit un questionnaire. Par exemple:



Pour ce questionnaire, il faut 2 ou 3 questions pour déterminer celle des cinq issues qui s'est produite.

- Le nombre moyen de questions à poser vaut $N = 2(0.3 + 0.2 + 0.2) + 3(0.15 + 0.15) = 2.3$.
- L'entropie de X vaut quant à elle $H(X) = 2.27$.

Pour ce questionnaire, le nombre moyen de questions à poser est très proche de l'entropie. Cette caractéristique provient de la bonne conception du questionnaire : chaque question est choisie de telle sorte que les deux réponses possibles aient approximativement la même probabilité, c'est-à-dire de manière à maximiser l'entropie. Dans ces conditions chacune des réponses apporte le maximum d'information.

Nous verrons que l'entropie est un minorant du nombre moyen de questions binaires à poser pour déterminer le résultat d'une expérience aléatoire.

La notion d'entropie est importante dans de nombreux domaines, le plus immédiat est celui de la compression : si dans un message long, les différents caractères utilisés pour écrire le message apparaissent avec des fréquences différentes les unes des autres, cela signifie que les caractères qui composent le message n'apportent pas autant d'information que cela est théoriquement possible. Une réécriture du message avec une loi uniforme sur l'alphabet utilisé permettra une compression sans perte, aussi appelée codage entropique.

Exemple de questionnaire optimal

On lance une pièce jusqu'à obtenir face. La probabilité de face est noté p, celle de pile $q = 1 - p$. L'entropie de la pièce (loi de Bernoulli) vaut $H = -p\log_2(p) - q\log_2(q)$.

Le nombre de lancers est une v.a. X à valeurs dans $\mathbb{N}^*$ de loi géométrique :

$$P(X = n) = pq^{n-1}, n \in \mathbb{N}^*$$

L'entropie de X s'écrit :

$$\begin{aligned}
H(X) &= -\sum_{n=1}^{+\infty} pq^{n-1}\log_2\left(pq^{n-1}\right) \\
&= -p\log_2(p)\sum_{n=1}^{+\infty} q^{n-1} - p\log_2(q)\sum_{n=1}^{+\infty} q^{n-1}(n-1) \\
&= -p\log_2(p)\underbrace{\sum_{n=1}^{+\infty} q^{n-1}}_{\frac{1}{1-q}=\frac{1}{p}} - p\log_2(q)\underbrace{\sum_{n=1}^{+\infty} nq^n}_{\frac{q}{(1-q)^2}=\frac{q}{p^2}} \\
&= \frac{-p\log_2(p) - q\log_2(q)}{p} \\
&= \frac{H}{p} \text{ bits.}
\end{aligned}$$

Pour $p = 1/2$, $H(X) = 2$ bits.

Pour trouver le résultat de cette expérience, le questionnaire suivant est efficace car les réponses à chacune des questions binaires sont de mêmes probabilités :

- $X = 1$?
  - Si, oui, terminé en 1 question (avec probabilité $1/2$)
  - Si non, $X = 2$?
    - Si, oui, terminé en 2 questions (avec probabilité $1/4$)
    - Si non, $X = 3$?
      - Si, oui, terminé en 3 questions (avec probabilité $1/8$)
      - Si non, $X = 4$? $\cdots$

Réponse en une question avec probabilité $1/2$, en 2 questions avec probabilité $1/4$, en 3 questions avec probabilité $1/8$, etc. La longueur moyenne du questionnaire est exactement égale à l'entropie de X :

$$\sum_{n=1}^{\infty} n\left(\frac{1}{2}\right)^n = \frac{\frac{1}{2}}{\left(1-\frac{1}{2}\right)^2} = 2 = H(X)$$

1.5 Propriétés de l'entropie

1.5.1 Propriétés démontrées en TD

Maximisantes et minimisantes de l'entropie d'une v.a. à état fini.

- $0 \leq H(p_1, \cdots, p_N) \leq \log_2 N$
  - $H(p_1, \cdots, p_N) = 0$ lorsque la v.a. X est déterministe. Une variable déterministe est d'entropie minimale (désordre minimum).

– $H(\frac{1}{N}, \cdots, \frac{1}{N}) = \log_2 N$: la loi uniforme est d'entropie maximale (désordre maximum). Pour un nombre d'états $N < +\infty$ l'entropie est maximale lorsque la distribution de probablité est uniforme sur l'ensemble des états. La loi uniforme discrète donne le même poids $1/N$ à chacune des N réalisations possibles. Du fait de la symétrie de la loi (par permutation des variables), l'incertitude moyenne (entropie) est égale à l'incertitude liée à chacune des réalisations. En particulier, lorsque $N = 2^k$, l'entropie est égale à k bits. On retrouve de cette façon l'usage courant de l'unité d'information. En informatique, un octet comporte 8 bits d'information. Nous voyons ici que cette affirmation n'est exacte, au sens de la théorie de l'information, que lorsque les "bits" sont des v.a. de Bernoulli équilibrées statistiquement indépendantes (la loi est uniforme sur les 256 états). Nous sommes ainsi amenés à séparer la notion de bit, au sens de valeur binaire, de celle de bit d'information.

Inégalité de Gibbs.

- Soient $P = \{p_i\}_{i\in\{1,\cdots,N\}}$ et $Q = \{q_i\}_{i\in\{1,\cdots,N\}}$ deux lois de probabilité.

$$\sum_{i=1}^{N} p_i \log_2 \frac{q_i}{p_i} \leq 0 \text{ avec égalité lorsque les 2 lois sont identiques} \tag{1.1}$$

1.5.2 Propriétés laissées à titre d'exercice

- $H(p_1, \cdots, p_N)$ est une fonction positive, symétrique et continue des variables p_i.
- $H(p_1, \cdots, p_N)$ est une fonction strictement concave des p_i sur l'ensemble des lois de probabilité.
- L'association de plusieurs événements fait décroître l'entropie. Cette propriété est claire sur le cas limite qui consiste à grouper en un seul état l'ensemble des valeurs possibles d'une v.a. X. Inversement, la dissociation d'évènements accroît l'entropie.

Chapter 2

Transinformation

Contents

2.1 Entropie conjointe

Considérons deux v.a. X à valeurs dans $x_1, \cdots, x_N$ et Y à valeurs dans $y_1 \cdots, y_M$. Il est possible de considérer le couple (X, Y) comme une seule v.a. Z pouvant prendre NM états $z_1, \cdots, z_{NM}$ en associant de manière bijective les z_k aux couples (x_i, y_j) par $z_{i+(j-1)N} = (x_i, y_j)$. D'où la définition naturelle $H(X,Y) = H(Z) = -\sum_{i=1}^{N}\sum_{j=1}^{M} p(x_i, y_j)\log_2 p(x_i, y_j)$.

2.1.1 Définition de l'entropie conjointe.

Pour deux v.a. X, à valeurs dans l'alphabet $\mathcal{A}_X$, et Y, à valeurs dans $\mathcal{A}_Y$, l'entropie conjointe des v.a. X et Y est définie par :

$$H(X,Y) = -\sum_{x\in\mathcal{A}_X}\sum_{y\in\mathcal{A}_Y} p(x,y)\log_2 p(x,y) \tag{2.1}$$

2.1.2 Majoration de l'entropie conjointe.

Comparaison de l'entropie de la loi composée à l'entropie des lois marginales :

$$H(X,Y) \leq H(X) + H(Y) \text{ (égalité lorsque } X \text{ et } Y \text{ sont indépendantes).} \tag{2.2}$$

En effet

$$H(X) + H(Y) = -\sum_{x\in\mathcal{A}_X} p(x)\log_2 p(x) - \sum_{y\in\mathcal{A}_Y} p(y)\log_2 p(y)$$

or,

$$\begin{aligned} p(x) &= \sum_{y\in\mathcal{A}_Y} p(x,y) \\ p(y) &= \sum_{x\in\mathcal{A}_X} p(x,y) \end{aligned}$$

d'où

$$H(X)+H(Y) = -\sum_{x\in\mathcal{A}_X}\sum_{y\in\mathcal{A}_Y} p(x,y)\log_2 p(x)p(y)$$

La comparaison à $H(X,Y) = -\sum_{x\in\mathcal{A}_X}\sum_{y\in\mathcal{A}_Y} p(x,y)\log_2 p(x,y)$ résulte de (1.1).

$$\underbrace{\left\{-\sum_{x\in\mathcal{A}_X}\sum_{y\in\mathcal{A}_Y} p(x,y)\log_2 p(x,y)\right\}}_{H(X,Y)} - \underbrace{\left\{-\sum_{x\in\mathcal{A}_X}\sum_{y\in\mathcal{A}_Y} p(x,y)\log_2 p(x)p(y)\right\}}_{H(X)+H(Y)}$$

$$=$$

$$\sum_{x\in\mathcal{A}_X}\sum_{y\in\mathcal{A}_Y} p(x,y)\log_2\frac{p(x)p(y)}{p(x,y)} \leq 0$$

soit

$$H(X,Y) \leq H(X)+H(Y)$$

avec égalité lorsque X et Y sont indépendantes. □

2.2 Entropie conditionnelle

L'entropie d'une v.a. Y est une fonction des probabilités *a priori* $p(y)$. Si l'état d'une autre v.a. X a été observé, l'incertitude moyenne sur Y sachant que $X = x$, que nous noterons indifféremment $H(Y|X=x)$ ou $H(Y|x)$, peut être définie par :

$$H(Y|X=x) = H(Y|x) = -\sum_{y\in\mathcal{A}_Y} p(y|x)\log_2 p(y|x) \tag{2.3}$$

Cette quantité représente l'entropie *a posteriori* sur Y sachant que X s'est réalisée en x.

2.2.1 Définition de l'entropie conditionnelle

De façon plus générale, il est intéressant de chiffrer l'entropie *a posteriori* moyenne qu'il est possible de définir de la manière suivante :

$$H(Y|X) = \sum_{x\in\mathcal{A}_X} p(x)H(Y|x) \tag{2.4}$$

En explicitant $H(Y|X=x)$, cette entropie conditionnelle s'écrit :

$$H(Y|X) = -\sum_{x\in\mathcal{A}_X}\sum_{y\in\mathcal{A}_Y} p(x,y)\log_2 p(y|x) \tag{2.5}$$

L'entropie conditionnelle de Y par rapport à X est donnée par l'espérance de $-\log_2 p(Y|X)$ par rapport à la loi conjointe de X et de Y.

2.2.2 Propriétés de l'entropie conditionnelle

Règle de chaînage.

$$H(X,Y) = H(X) + H(Y|X) = H(Y) + H(X|Y) \tag{2.6}$$

En effet

$$\begin{aligned} H(X,Y) &= -\sum_{x\in\mathcal{A}_X}\sum_{y\in\mathcal{A}_Y} p(x,y)\log_2 \overbrace{p(x,y)}^{p(x)p(y|x)} \\ &= -\sum_{x\in\mathcal{A}_X}\sum_{y\in\mathcal{A}_Y} p(x,y)\log_2 p(x) - \sum_{x\in\mathcal{A}_X}\sum_{y\in\mathcal{A}_Y} p(x,y)\log_2 p(y|x) \\ &= H(X) + H(Y|X) \end{aligned}$$

□

L'extension à n variables est immédiate :

$$H(X_1,\cdots,X_n) = \sum_{i=1}^{n} H(X_i|X_{i-1},\cdots,X_1) \tag{2.7}$$

Réduction de l'entropie par conditionnement.

De

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

on déduit la majoration de l'entropie conditionnelle par l'entropie *a priori* :

$$H(Y|X) \leqslant H(Y) \tag{2.8}$$

La variable X apporte de l'information et réduit l'incertitude sur Y. L'entropie *a posteriori* sur Y (après observation de X) est plus faible que l'entropie *a priori* (avant observation de X) .

Attention, ceci est un résultat en moyenne qui n'est pas exact pour le conditionnement par un évènement donné.

Deux cas particuliers utiles :

v.a. totalement dépendantes $Y = X$. $\Pr(X = \alpha|X = x) = \delta_{x-\alpha}$ (1 si $x = \alpha$ et 0 sinon), l'entropie de cette loi déterministe est nulle $H(X|x) = 0$, d'où $H(X|X) = \sum_{x\in\mathcal{A}_X} p(x)H(X|x) = 0$. Sachant X, il ne subsiste aucune incertitude sur X.

v.a. totalement indépendantes X et Y indépendantes. $H(X)+H(Y) = H(X,Y) = H(X)+H(Y|X)$ d'où $H(Y) = H(Y|X)$. Sachant X (indépendante de Y) l'incertitude sur Y est toujours $H(Y)$.

Asymétrie de l'entropie conditionnelle.

L'entropie conditionnelle n'est pas symétrique :

$$\left.\begin{array}{rcl} H(Y|X) &=& H(X,Y) - H(X) \\ H(X|Y) &=& H(X,Y) - H(Y) \end{array}\right\} \Rightarrow H(Y|X) \neq H(X|Y)$$

2.3 Divergence de Kullback et information mutuelle

2.3.1 Divergence de Kullback-Leibler ou entropie relative.

La divergence de Kullback, ou entropie relative, entre deux distributions p et q définies sur un même alphabet $\mathcal{A}$ est donnée par :

$$D(p||q) = \sum_{x\in\mathcal{A}} p(x)\log_2\frac{p(x)}{q(x)} \tag{2.9}$$

On adopte les conventions naturelles $0 \log_2 \frac{0}{0} = 0$, $0 \log_2 \frac{0}{q} = 0$ et $p \log_2 \frac{p}{0} = \infty$. D'après (1.1) la divergence de Kullback est non négative :

$$D(p||q) \geq 0 \tag{2.10}$$

avec égalité si et seulement si les lois p et q sont identiques.

Cette divergence mesure une proximité entre deux lois mais ce n'est pas une distance : elle n'est pas symétrique et ne satisfait pas l'inégalité triangulaire.

2.3.2 Information mutuelle.

L'information mutuelle (notée $I(X;Y)$) est la divergence de Kullback entre la loi conjointe $p(x,y)$ et le produit de ses marginales $p(x)p(y)$:

$$I(X;Y) = \sum_{x \in \mathcal{A}_X} \sum_{y \in \mathcal{A}_Y} p(x,y) \log_2 \frac{p(x,y)}{p(x)p(y)}$$

Elle mesure l'information apportée par une v.a. sur une autre. L'observation d'une v.a. Y réduit l'incertitude moyenne sur toute v.a. X statistiquement liée à Y. $I(X;Y)$ est la réduction d'incertitude sur X due à la connaissance de Y, on a :

$$I(X;Y) = H(X) - H(X|Y) \tag{2.11}$$

Preuve :

$$\begin{aligned} I(X;Y) &= \sum_{x \in \mathcal{A}_X} \sum_{y \in \mathcal{A}_Y} p(x,y) \log_2 \frac{\overbrace{p(x,y)}^{p(x|y)p(y)}}{p(x)p(y)} \\ &= \left\{ \underbrace{- \sum_{x \in \mathcal{A}_X} \sum_{y \in \mathcal{A}_Y} p(x,y) \log_2 p(x)}_{H(X)} \right\} - \left\{ \underbrace{- \sum_{x \in \mathcal{A}_X} \sum_{y \in \mathcal{A}_Y} p(x,y) \log_2 p(x|y)}_{H(X|Y)} \right\} \end{aligned}$$

□

Si $Y = X$, l'information mutuelle entre une v.a. X et elle-même se réduit à l'entropie :

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

Une propriété intéressante de cette information mutuelle est sa symétrie : l'information apportée par la v.a. Y sur la v.a. X est égale à la réduction d'incertitude moyenne sur X due à l'observation de Y.

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

De la règle de chaînage de l'entropie conditionnelle (2.7), on déduit de manière directe celle de l'information mutuelle :

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

Chapter 3

Compressibilité et entropie.

Contents

Lorsque l'on cherche à réécrire un message dans un alphabet donné de la manière la plus brève possible, une question essentielle est la détermination d'une borne inférieure à cette compacité. L'existence d'une telle borne permet de situer les performances de tout algorithme pratique de compression par rapport à cette performance optimale.

L'objectif principal de ce chapitre est de trouver une telle borne en montrant que la longueur moyenne minimale du message codé est liée à l'entropie.

3.1 Définition d'un codage de source

Soit X une v.a. prenant ses valeurs x dans l'ensemble fini $\mathcal{A}_X = \{x_1, \cdots, x_N\}$ avec les probabilités $\{p(x_1) = p_1, \cdots, p(x_N) = p_N\}$.

3.1.1 Codage d'un seul état

Définition d'un code source. Un code source C est une application de $\mathcal{A}_X$ dans $\mathcal{D}^*$ où $\mathcal{D}^*$ est l'ensemble des séquences de longueur finie écrites dans un alphabet à D caractères $\{a_1, \cdots, a_D\}$.

$$\begin{array}{rccc} C: & \mathcal{A}_X & \to & \mathcal{D}^* \\ & x & \to & C(x) \end{array}$$

Le code source C associe le mot de code $C(x)$ à l'état x de la v.a. X.

Ainsi, chacun des N états possibles $x_1, \cdots, x_N$ (alphabet de la source) de la v.a. X est décrit à l'aide d'une séquence finie de caractères de l'alphabet du code $a_1, \cdots, a_D$. Chacune de ces séquences s'appelle un mot-code ; l'ensemble des mots du code s'appelle le code.

Remarque. Ce problème est le même que celui de la théorie des questionnaires (évoqué au chapitre 1). Les caractères des mots du code peuvent être considérés comme les réponses à des questions destinées à déterminer le résultat de l'expérience aléatoire.

Compacité. La longueur du mot de code $C(x)$ est notée $l(x)$ (ou $l(x_j) = l_j$). La longueur moyenne des mots, aussi appelée compacité du code, s'écrit :

$$\nu = \sum_{x \in \mathcal{A}_X} p(x) l(x) = \sum_{j=1}^{N} p_j l_j$$

L'espérance est prise par rapport à la loi de X : les symboles peu probables n'accroissent que peu la taille moyenne des messages codés alors que les symboles très probables contribuent fortement à la longueur moyenne et doivent de ce fait être les plus courts possible.

Code non singulier. Un code est dit non singulier si l'application C est injective, c'est-à-dire si deux états différents sont codés par deux séquences différentes ; autrement dit si un même mot de code ne peut pas correspondre à plusieurs états de la v.a. X.

Pour un code non singulier, il est possible de remonter sans ambiguïté d'un mot du code à la valeur encodée.

3.1.2 Codage d'un message composé d'une suite d'états

Source simple. Une source faite de copies indépendantes d'une même v.a. X est dite simple. Une telle source délivre des messages composés de réalisations indépendantes d'une v.a. X[1].

Dans la suite, seules des sources simples sont considérées.

Message source. Une suite de n états de la source constitue un message, c'est-à-dire une suite de caractères appartenant à l'alphabet de la source (qui compte N caractères $x_1, \cdots, x_N$).

Codage d'une source simple. L'objectif du codage de source est de transformer un message source en un message codé, c'est-à-dire en une suite de mots du code, eux-mêmes composés de caractères appartenant à l'alphabet[2] du code $a_1, \cdots, a_D$.

Extension du codage C d'un unique état au codage C^* d'un message source. On note $x_{i_k} \in \{x_1, \cdots, x_N\}$ le $k^{\text{ème}}$ état généré par la source, de sorte qu'un message source de longueur n s'écrit $x_{i_1} \cdots x_{i_n}$.

L'extension du codage C d'un unique état de la source en un codage C^* d'un message composé de n caractères (suite de n états de la source) est telle que :

$$\begin{array}{rccc} C^* : & (\mathcal{A}_X)^n & \longrightarrow & \mathcal{D}^* \\ & \underbrace{x_{i_1} \cdots x_{i_n}} & \longrightarrow & \underbrace{C(x_{i_1}) \cdots C(x_{i_n})} \\ & \text{Message source} & & \text{Message codé} \end{array}$$

$C(x_{i_1}) \cdots C(x_{i_n})$ désigne la concaténation des mots $C(x_{i_1})$ à $C(x_{i_n})$.

Codes déchiffrables. Un code est déchiffrable si à chaque message codé correspond au plus un message source. Pour que tout message codé soit déchiffrable, il faut que C^* soit non singulier. Un code déchiffrable est aussi qualifié de séparable du fait que la déchiffrabilité résulte de la possibilité de séparer les mots successifs à la lecture du message codé.

[1] Des sources plus complexes introduisent des dépendances entres les valeurs générées.

[2] Cet alphabet correspond par exemple au type de lettres qu'il est possible de stocker sur un support (0 ou 1 dans une mémoire informatique par exemple) ou aux lettres qu'il est possible de transmettre au travers d'un canal donné.

Assurer la déchiffrabilité. Les méthodes classiques qui permettent d'assurer la déchiffrabilité sont :

Longueur unique des mots. La façon la plus évidente d'assurer la déchiffrabilité d'un code consiste à attribuer la même longueur à tous les mots-code. Ainsi, il suffit de compter les caractères pour séparer les mots du code. Les codes ASCII et UTF, par exemple, utilisent cette procédure. Celle-ci peut être naturelle lorsque les données sont stockées par paquet (typiquement octet ou ensemble d'octets).

Séparateur. Une autre manière de procéder consiste à utiliser un caractère supplémentaire destiné à identifier la fin d'un mot-code. Cette technique dégrade bien évidemment les performances du code. Exemple : en morse, un intervalle de temps est placé entre les lettres.

Condition du préfixe. Une condition suffisante de déchiffrabilité plus intéressante, appelée condition du préfixe, consiste à imposer aux mots d'un code la propriété suivante : aucun mot-code ne doit être le préfixe d'un autre mot-code.

Les codes qui vérifient la condition du préfixe sont dit instantanés (ou parfois irréductible). Cette appellation est due au fait qu'il est possible d'effectuer le décodage pas à pas (sans avoir à attendre d'autres caractères pour prendre une décision quant au mot-code lu) : dès qu'un mot est reconnu, il est possible de le séparer de la suite du message sans attendre de lire les caractères qui suivent. Les codes instantanés ont une grande importance pratique.

Un code instantané est déchiffrable mais tous les codes déchiffrables ne sont pas instantanés. A titre d'exemple, considérons le code $\{M_1 = 0, M_2 = 00001\}$. Ce code n'est pas instantané puisque M_1 est le préfixe de M_2. Il est néanmoins déchiffrable. A lecture du message 000001, il n'est pas possible de décider que le premier 0 correspond au mot M_1 puisque ce 0 peut aussi n'être que le début de M_2. Cependant, à la lecture du cinquième 0, on sait de façon certaine que le premier 0 correspondait à M_1. Pour décider M_1 il a fallu attendre pour s'assurer qu'il ne s'agissait pas d'un autre mot-code. C'est ici que se situe l'origine du terme non instantané.

3.2 CNS d'existence d'un code instantané

La taille N de l'alphabet de la source (nombre d'états de la v.a. X), la taille D de l'alphabet du code et les longueurs $\{l_i = l(x_i)\}_{i=1\cdots N}$ des mots du code étant fixées, il n'est pas toujours possible de construire un code. Les mots du code doivent avoir une longueur suffisante pour permettre de représenter toutes les éventualités et satisfaire la contrainte de déchiffrabilité. Dans le cas des codes instantanés, une CNS d'existence est donnée par l'inégalité de Kraft.

Inégalité de Kraft. Si D désigne la taille de l'alphabet utilisé pour le codage, un code instantané composé de mots de longueurs $l_1, l_2 \cdots, l_N$ existe si et seulement si

$$\sum_{i=1}^{N} D^{-l_i} \leqslant 1$$

Notons $l_1 \leq l_2 \leq \cdots \leq l_N$ la longueur des mots-code associés aux états x_i de X. Le nombre maximum de mots de longueur l_N qu'il est possible de former à l'aide d'un alphabet à D lettres est de D^{l_N}.

Supposons qu'il existe un code instantané, l'utilisation du mot de longueur l_i élimine $D^{l_N - l_i}$ possibilités (condition du préfixe). On ne peut pas éliminer plus de point terminaux qu'il n'en existe, c'est-à-dire $\sum_{i=1}^{N} D^{l_N - l_i} \leqslant D^{l_N}$, l'inégalité de Kraft en découle en divisant par D^{l_N}.

Réciproquement, si l'inégalité de Kraft est vérifiée, il est possible de construire un code instantané en choisissant successivement les mots de longueurs $l_1, l_2, \cdots, l_N$.

□

Ce résultat s'étend à l'ensemble des codes déchiffrables, il porte alors le nom de théorème de Mac Millan. Cette extension est d'une grande importance pratique et montre que l'on peut se cantonner à utiliser des codes

instantanés sans être pénalisé du point de vue des performances en compression. En effet, s'il existe un code déchiffrable avec des mots de longueurs l_i, ce code satisfait la condition de Mac-Millan et il existe un code instantané pour les mêmes longueurs l_i.

3.3 Codes optimaux théoriques.

Ce paragraphe démontre deux résultats qui permettront ensuite de construire effectivement des codes optimaux :

1. Il existe une borne inférieure pour la longueur moyenne des mots d'un code.

2. Il est possible de s'approcher aussi près qu'on le souhaite de cette borne.

3.3.1 Efficacité d'un code.

On cherche à compresser au mieux un message sans perdre d'information, c'est-à-dire en étant capable de restituer exactement le message original à partir de sa version compressée. Cela revient à réécrire le message avec un nombre de caractères-code aussi réduit que possible. Pour cela, un des critères possibles est celui de la longueur moyenne minimale pour les mots qui composent le code.

La mesure d'incertitude H permet de caractériser le taux de compression qu'il est possible d'espérer pour un message donné et rend possible une mesure effective de l'efficacité d'une procédure de codage.

L'entropie de la source d'information étant $H(X)$ et la longueur moyenne des mots-code ν ; l'entropie moyenne par caractère-code (après codage) est égale à $H(X)/\nu$, cette quantité est majorée par l'entropie de la loi uniforme sur l'ensemble des caractères du code. Ainsi $H(X)/\nu \leq \log_2(D)$. Un code est d'autant plus efficace qu'il approche cette borne, ceci conduit à définir l'efficacité d'un code par :

$$E = \frac{H(X)}{\nu \log_2(D)}$$

Des conditions d'optimalité commencent à apparaître : si ν est un entier, il existe D^ν mots de longueur ν dont les caractères appartiennent à un alphabet de taille D. L'information est maximale lorsque les mots sont équiprobables, elle vaut alors $\log_2[D^\nu] = \nu \log_2[D]$. Cette borne supérieure est atteinte lorsque les caractères qui composent le mot sont indépendants et que chacun d'eux est distribué uniformément sur les D caractères de l'alphabet du code.

Les deux racines de redondance sont là : dépendance entre caractères et distribution non uniforme.

3.3.2 1er théorème de Shannon : la compacité est minorée par l'entropie

Une caractéristique essentielle d'un code réside dans la longueur moyenne des mots qu'il emploie, sa compacité. Cette compacité dépend de la nature de la source (des messages) à coder. Lorsque la source délivre toujours le même message (l'un des symboles est de probabilité un), son entropie est nulle, et le codage est immédiat (un seul mot code, le plus court possible, *i.e.* de longueur 1). Lorsque la source délivre des symboles équiprobables, l'entropie est maximale et l'on conçoit qu'il n'est pas de compression possible. La prise en considération de ces cas extrêmes fait pressentir le rôle de l'entropie de la source sur la compacité du code. Nous allons maintenant formaliser cette intuition. Le résultat obtenu, appelé premier théorème de Shannon, montre qu'il existe une borne inférieure pour la compacité et précise que cette borne peut être atteinte asymptotiquement.

On note $H_D(X)$ l'entropie en base D de la v.a. X :

$$H_D = -\sum_{i=1}^{N} p_i \log_D p_i = \frac{H(X)}{\log_2 D}$$

H_D est une borne inférieure pour la compacité ($\nu \geqslant H_D(X)$)

La borne de compacité maximale est atteinte pour $p_i = D^{-l_i^*} \leftrightarrow l_i^* = -\log_D p_i$

Ecriture de la preuve à l'aide de la divergence de Kullback.

Calculons la différence entre la longueur moyenne des mots du code et l'entropie :

$$\nu - H_D(X) = \sum_{i=1}^{N} p_i l_i - \left(-\sum_{i=1}^{N} p_i \log_D p_i \right)$$

en insérant $l_i = -\log_D D^{-l_i}$:

$$\begin{aligned}
\nu - H_D(X) &= \sum_{i=1}^{N} p_i \log_D p_i - \sum_{i=1}^{N} p_i \log_D D^{-l_i} \\
&= \sum_{i=1}^{N} p_i \log_D \left[\frac{p_i}{\left(\frac{D^{-l_i}}{\sum_{i=1}^{N} D^{-l_i}} \right) \sum_{i=1}^{N} D^{-l_i}} \right] \\
&= \underbrace{\log_D \frac{1}{\sum_{i=1}^{N} D^{-l_i}}}_{\geq 0} + \underbrace{\sum_{i=1}^{N} p_i \log_D \left[\frac{p_i}{\left(\frac{D^{-l_i}}{\sum_{i=1}^{N} D^{-l_i}} \right)} \right]}_{D(p||q) \geq 0}
\end{aligned}$$

Le second terme est la divergence de Kullback $D(p||q)$ entre la loi $\{p_i\}_{i=1\cdots N}$ et la loi $\left\{ q_i = \frac{D^{-l_i}}{\sum_{i=1}^{N} D^{-l_i}} \right\}_{i=1\cdots N}$ tandis que le premier terme est positif ou nul d'après l'inégalité de Kraft. On en déduit :

$$\nu - H_D(X) \geq 0$$

L'égalité est atteinte si et seulement si les deux conditions suivantes sont vérifiées :

1. $D(p||q) = 0$, c'est-à-dire $p_i = \frac{D^{-l_i}}{\sum_{i=1}^{N} D^{-l_i}}$
2. $\sum_{i=1}^{N} D^{-l_i} = 1$

Autrement dit $\nu = H_D(X)$ si et seulement si $p_i = D^{-l_i}$.

Ceci est possible si et seulement si $l_i = -\log_D p_i$ est entier pour tout $i \in \{1 \cdots N\}$. Ce ceci se produit lorque la distribution des probabilités est de la forme $p_i = D^{-n_i}$ avec n_i entier, une telle distribution est dite D-adic.

□

En pratique, la borne est atteinte à au plus 1 bit près.

Le choix $p_i = D^{-l_i}$ conduit aux longueurs $l_i = -\log_2(p_i)/\log_2(D)$ non-nécessairement entières. Le mieux qu'il est possible de faire est d'arrondir à l'entier supérieur, ainsi :

$$-\log_2(p_i)/\log_2(D) < l_i < -\log_2(p_i)/\log_2(D) + 1$$

En multipliant par p_i et en sommant sur l'ensemble des symboles, on obtient le meilleur encadrement pour la compacité :

$$H_D(X) \leqslant \nu \leqslant H_D(X) + 1$$

Ce résultat dit simplement que le cas le plus défavorable est celui pour lequel tous les mots ont une longueur optimale très légèrement supérieure à un entier.

Remarque. Nous avons ainsi démontré que l'entropie en base D représente le nombre moyen minimum de questions qu'il est nécessaire de poser pour déterminer le résultat d'une expérience.

3.4 Codage par bloc

Pour approcher plus finement la borne inférieure, l'idée est de ne plus coder individuellement chacun des états possibles de la source mais de grouper les symboles par bloc pour ensuite coder ces blocs.

Ce groupement peut être vu de la façon suivante.

3.4.1 Extension d'ordre s de la source X.

On appelle extension d'ordre s de la source X la source qui groupe s v.a. successives de type X : $Y = (X_1, \cdots, X_s)$.

L'extension d'ordre s d'une source simple est également une source simple, cette extension peut prendre N^s états.

Par exemple le message $x_{i_1}, \cdots, x_{i_s}, x_{i_{s+1}}, \cdots, x_{i_{2s}}$ issu de X correspond au message y_{i_1}, y_{i_2} issu de Y avec $y_{i_1} = x_{i_1}, \cdots, x_{i_s}$ et $y_{i_2} = x_{i_{s+1}}, \cdots, x_{i_{2s}}$.

3.4.2 Le codage par bloc permet de mieux approcher la borne inférieur H_D.

L'encadrement de la compacité pour l'extension d'ordre s de la source X s'écrit :

$$H_D(Y) \leqslant \nu_s \leqslant H_D(Y) + 1$$

où ν_s est la longueur moyenne des mots pour la source Y, c'est-à-dire pour un groupe de s états de la source X.

Les symboles délivrés par la source X étant indépendants (source simple), l'entropie de $(X_1, \cdots, X_s)$ est la somme des entropies des composantes X_j, les X_j étant des copies indépendantes de X, on a $H(Y) = sH(X)$ d'où

$$H_D(X) \leqslant \nu \leqslant H_D(X) + \frac{1}{s}$$

En codant des extensions de plus en plus longues de la source, il est possible d'approcher la borne optimale. La borne est approchée en introduisant un retard (il faut attendre que s symboles sources soient réunis avant d'effectuer le codage) et en augmentant la complexité du codeur.

3.4.3 Exemple — illustration du gain d'un codage par bloc

Soit X une source qui délivre deux symboles $x_1 =' A'$ et $x_2 =' B'$ avec les probabilités $p(A) = 0.8$ et $p(B) = 0.2$.

L'entropie de la source vaut

$$H(X) = -0.8 \log_2(0.8) - 0.2 \log_2(0.2) = 0.72 \text{ bits}$$

Codage direct de la source. Les longueurs de mot optimales, au sens de la compacité, $l_1^* = -\log_2 0,8 = 0,32$ et $l_2^* = -\log_2 0,2 = 2,32$ permettent d'atteindre la borne inférieure (entropie de la source), elles sont non entières : longueur très courte pour le mot le plus probable, plus longue pour l'autre mot. Arrondir à l'entier supérieur (1 et 3) ne permet pas de construire le meilleur code : le mot de longueur 3 est inutilement long puisqu'il suffit de le choisir de longueur 1 avec comme caractère de code celui qui n'est pas utilisé par le mot de longueur 1 qui code l'état 'A' : dans le tableau ci-dessous, l'état 'A' de la source est codé par le mot 1 et l'état 'B' par le mot 0, la longueur moyenne des mots du code est de $\nu = 1$ caractère.

Finalement, le codage est évident : la source possède 2 états, les mots du code sont choisis aussi courts que possible (longueur 1) et l'alphabet le plus réduit possible ($D = 2$ caractères $a_1 =' 0'$ et $a_2 =' 1'$).

Les longueurs des mots diffèrent des longueurs optimales $l_i^* = -\log_2 p_i$. On vérifie l'encadrement :

$$0.72 = H(X) \leqslant \nu = 1 \leqslant H(X) + 1 = 1.72$$

Symbole	$p(x_i)$	l_i^*	l_i	Mots
A	0.8	0.32	1	1
B	0.2	2.32	1	0

On constate, sur ce cas particulier, que l'impossibilité de choisir des mots de longueur non entière augmente la longueur moyenne de moins de 1 caractère (0.28 dans cet exemple, soit un surcoût non négligeable de 39% sur la longueur moyenne des messages codés).

Codage des extensions d'ordre 2 de la source. Les longueurs optimales des mots ($l_1^* = -\log_2 0,8 = 0,32$ et $l_2^* = -\log_2 0,2 = 2,32$) laissent penser qu'un codage par bloc est intéressant. Considérons l'extension d'ordre $s = 2$ de la source, c'est-à-dire la v.a. Y pouvant prendre les quatre états $y_1 =' AA'$, $y_2 =' AB'$, $y_3 =' BA'$ et $y_4 =' BB'$ avec les probabilités $p_{AA} = p(A)p(A)$, $p_{AB} = p_{BA} = p(A)p(B)$ et $p_{BB} = p(B)p(B)$. On peut alors, par exemple, construire le code suivant :

Symbole	$p(y_i)$	l_i^*	l_i	Mots
AA	0.64	0.643	1	0
AB	0.16	2.643	2	10
BA	0.16	2.643	3	110
BB	0.04	4.643	3	111

Le codage des extensions d'ordre deux de la source conduit à la longueur moyenne $\nu_2 = 1.56$ caractère par paire de caractères de X. Autrement dit $\nu = 0.78$ caractère par symbole de la source X. On vérifie à nouveau l'inégalité :

$$0.72 = H(X) \leqslant \nu = 0.78 \leqslant H(X) + 1/2 = 1.22$$

On constate sur cet exemple l'efficacité d'un codage bloc : la perte est maintenant inférieure à $1/2$, ici elle vaut 0.06 (contre 0.28 pour le codage direct de la source) soit un surcoût de seulement 8% en terme de longueur moyenne des messages codés contre 39% pour le codage directe de la source.

Remarque :

3.4.4 Codage de sources non simples.

Le codage des extensions est certes utile pour que la compacité approche la borne H_D lors du codage d'une source simple mais, le gain reste en général faible et ce d'autant plus que les mots sont longs en moyenne. En pratique, il est le plus souvent inutile de coder des extensions d'ordre $s > 2$.

En revanche, le codage par bloc apporte des gains très importants pour le codage de sources non simples, il s'agit d'une façon simple et efficace de prendre en compte la dépendance des caractères générés par la source. En français par exemple, les lettres forment des syllabes composées 2 ou 3 lettres, ainsi le codage des extensions d'ordre 3 intègre au moins partiellement cette structure forte des messages et la compression croît considérablement (puisque la distribution des probabilités est très inégale sur les suites de 3 lettres, aaa,ztz n'apparaissent jamais par exemple, alors que tre est une sous séquence fréquente)

Chapter 4

Algorithmes de compression

Contents

On admet le résultat suivant : si un code instantané est optimal alors il est également optimal dans la classe (plus grande) des codes déchiffrables. C'est la raison pour laquelle nous nous restreignons aux seuls codes instantanés.

Les algorithmes sont présentés par défaut dans le cas binaire ($D = 2$), mais peuvent être généralisés au cas $D > 2$ ($D = 3$, code ternaire par exemple).

4.1 Conditions nécessaires d'optimalité

Avant de présenter les deux codes classiques que sont le code de Huffman et le code de Fano-Shannon, donnons quelques conditions nécessaires d'optimalité.

Soit C un code dont les mots ont les longueurs l_i avec les probabilités $p_1 \geq \cdots \geq p_N$ (Lorsque plusieurs mots ont la même probabilité, on les classe par ordre de taille croissante). Les conditions suivantes sont nécessaires pour que le code soit optimal :

1. $p_j > p_k \rightarrow l_j \leq l_k$.

   Si cette condition n'est pas vérifiée, il suffit de permuter les mots l_j et l_k pour obtenir un code plus performant et C ne serait pas optimal.

   Preuve. Si C est un code optimal (de compacité ν) et C' le code (de compacité ν') obtenu en permutant les mots j et k, on a :

   $$\begin{aligned} l'_i &= l_i \text{ pour } i \neq j, i \neq k \\ l'_j &= l_k \\ l'_k &= l_j \end{aligned}$$

   d'où d'où

   $$\nu' - \nu = (p_j - p_k)(l_k - l_j)$$

   D'après les deux hypothèses $p_j > p_k$ ($p_j - p_k > 0$) et C optimal ($\nu' - \nu \geq 0$), $\nu' - \nu \geq 0$ on a $l_k \geq l_j$ □

2. $l_N = l_{N-1}$: les deux mots les plus longs ont la même longueur.

3. Parmi les mots de longueur l_N, au moins deux ne diffèrent que par le dernier caractère puisque s'il n'en est pas ainsi, la suppression de celui-ci, conduit à un code plus performant.

4.2 Code optimal de Huffman et code de Fano-Shannon

Deux codes classiques :

Le code de Fano-Shannon. Très naturelle, la construction du code de Fanon-Shannon reprend l'idée déjà utilisée pour la construction de bons questionnaires. La loi uniforme maximise l'entropie, pour déterminer laquelle des N réalisations possibles de la source s'est effectivement produite, imaginons que l'on pose des questions binaires (deux réponses possibles). Le meilleur choix, pour une question donnée, consiste à équilibrer la probabilité des deux réponses possibles, ce choix maximise l'information. Telle est l'idée du codage de Fano-Shannon : diviser l'ensemble des réalisations possibles de la source en deux sous-ensembles de probabilités aussi voisines que possible puis renouveler l'opération sur chacun des sous-ensembles.

Le code de Huffmann. L'idée du code de Huffmann consiste à grouper les deux événements les moins probables en un unique événement et à renouveler l'opération avec le nouvel ensemble d'événements ainsi obtenu.

L'algorithme de Huffman (1952) est un algorithme glouton, c'est-à-dire un algorithme qui enchaîne des procédures localement optimales en vue d'un résultat global optimal. Cet algorithme construit un code instantané optimal. En pratique, il conduit à des réductions de longueur de l'ordre de 20 à 90% et il peut être associé à d'autres formes de codage.

Il modélise le code par une forêt composée d'arbres qui possèdent des noeuds dont le poids est la somme des poids de leurs enfants (le poids d'un arbre est celui de sa racine).

Initialement, les arbres (à 1 seul noeud) sont les états de la source, le poids de chacun de ces arbres est la probabilité de l'état associé.

A chaque étape, l'algorithme groupe deux des arbres de plus faible poids en un arbre unique dont ils sont les enfants.

Au final, l'algorithme produit un arbre unique de poids 1.

Non optimalité du code Fano-Shannon, un contre-exemple.

Considérons une source X à 4 états de probabilités $\pi_1^4 = 0.45, \pi_2^4 = 0.4, \pi_3^4 = 0.1, \pi_4^4 = 0.05$.

L'entropie de la source vaut :

$$\begin{aligned} H(X) &= -\sum_{i=1}^{4} \pi_i^4 \log_2 \pi_i^4 \\ &= -0.45 \log_2 0.45 - 0.4 \log_2 0.4 - 0.1 \log_2 0.1 - 0.05 \log_2 0.05 \\ &= 1.6 \text{ bit} \end{aligned}$$

Aucun code ne peut avoir une longueur moyenne inférieure à 1.6 caractères par mot. Nous allons vérifier sur cet exemple que le code de Fano-Shannon est sous-optimal.

Les longueurs optimales théoriques valent :

$$\begin{aligned} l_1^* &= 1.15 \\ l_2^* &= 1.32 \\ l_3^* &= 3.32 \\ l_4^* &= 4.32 \end{aligned}$$

En arrondissant à l'entier supérieur le jeu de longueurs devient $l_1 = 2, l_2 = 2, l_3 = 4, l_4 = 5$, soit une longueur moyenne de $0.45 \times 2 + 0.4 \times 2 + 0.1 \times 4 + 0.05 \times 5 = 2.35$.

Le code de Fano-Shannon (figure 4.1) n'a que des mots de longueur 2, sa longueur moyenne vaut $\nu_{FS} = 2$. Il est déjà meilleur qu'un simple arrondi à l'entier supérieur.

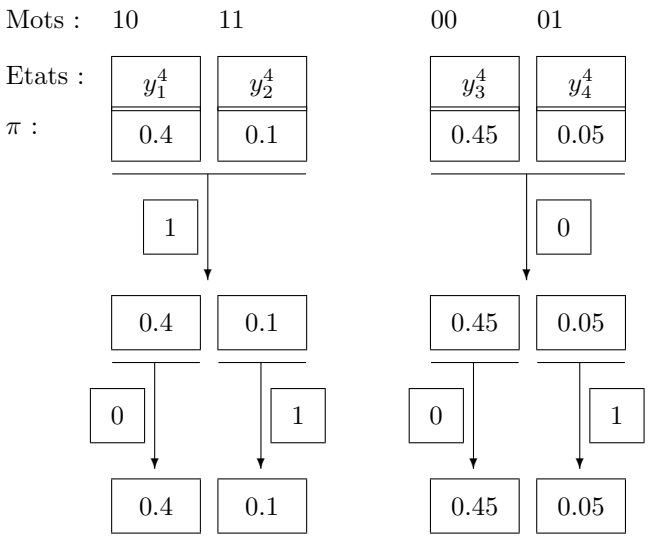


Figure 4.1: Codage de la source de jeu de probabilités $\pi_1^4 = 0.45, \pi_2^4 = 0.4, \pi_3^4 = 0.1, \pi_4^4 = 0.05$. Le code de Fano-Shannon est sous optimal, sa longueur moyenne vaut $\nu_{FS} = 2$.

Considérons maintenant le code de Huffman (figure 4.2), ses mots sont de longueurs variables, $\nu_H = 0.45 \times 1 + 0.4 \times 2 + 0.1 \times 2 + 0.05 \times 2 = 1.7$: $\nu_H < \nu_{FS}$, le code de Huffman est meilleur que celui de Fano-Shannon.

Ce contre exemple démontre que la procédure de Fano-Shannon n'est pas optimale.

Optimalité du code de Huffman.

En cours, on montre que la réduction de Huffman, qui permet de passer de n à $n-1$ états, correspond à une variation de longueur moyenne égale à la somme des probabilités des deux états fusionnés. Ainsi, le groupement des deux états les moins probables donne la variation de longueur la plus faible possible.

On montre également que l'optimalité au rang $n-1$ équivaut à l'optimalité au rang n et que le code optimal pour deux états consiste à coder l'un d'eux par le mot 0 et l'autre par 1.

Ces résultats démontrent l'optimalité du codage de Huffman.

4.3 Autres types de codes — code arithmétique

Ce paragraphe présente une autre méthode de compression : le codeur arithmétique.

On considère $x_{i_1} \cdots x_{i_n}$ (avec $i_j \in \{1, \cdots, N\}$), la séquence de longueur n à coder .

Le codeur arithmétique procède de la manière suivante.

Intervalle initial. On réalise tout d'abord une partition de l'intervalle $[0; 1)$ en N sous intervalles $I_j = [\alpha_j; \beta_j)$ de longueurs p_j, $j = 1 \cdots N$.

Intervalle courant. L'algorithme procède à un codage par intervalle au cours duquel l'intervalle initial est réduit en fonction de la suite des symboles à coder.

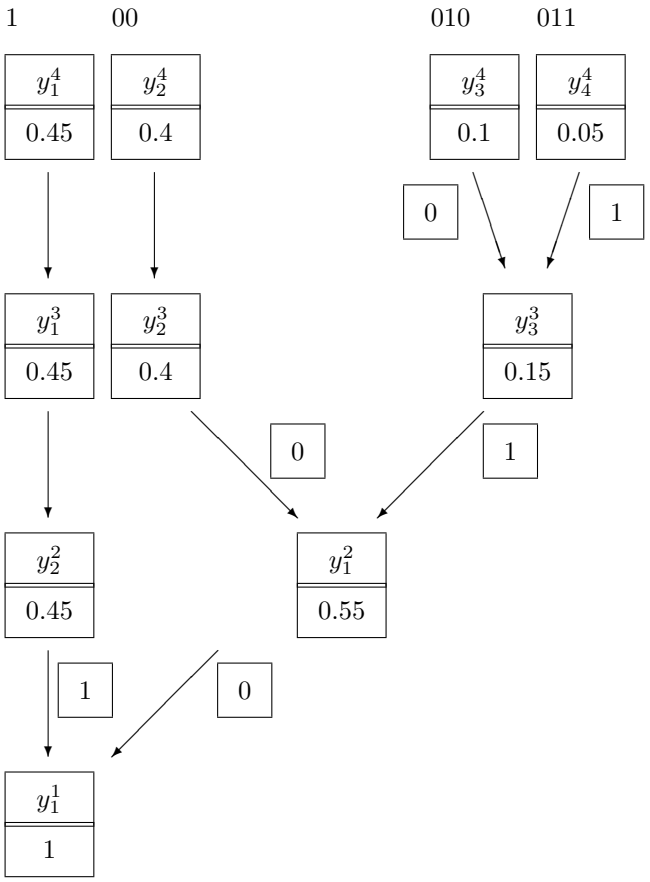


Figure 4.2: Codage de la source de jeu de probabilités $\pi_1^4 = 0.45, \pi_2^4 = 0.4, \pi_3^4 = 0.1, \pi_4^4 = 0.05$. Le code de Huffman est optimal, sa longueur moyenne vaut $\nu_H = 0.45 \times 1 + 0.4 \times 2 + 0.1 \times 2 + 0.05 \times 2 = 1.7$.

L'intervalle courant $[a_k; b_k)$ est de longueur notée $w_k = b_k - a_k$.
Initialement $a_0 = 0$, $b_0 = 1$ et $w_0 = 1$.

Codeur. Encoder le $(k+1)$ème caractère $x_{i_{k+1}}$ $(k \geq 0)$ (associé à l'intervalle $I_{i_{k+1}} = [\alpha_{i_{k+1}}; \beta_{i_{k+1}})$) se fait par réduction de l'intervalle courant $C_k = [a_k; b_k)$ en $[a_{k+1}; b_{k+1})$ selon :

$$a_{k+1} = a_k + w_k \, \alpha_{i_{k+1}} \tag{4.1}$$
$$b_{k+1} = a_k + w_k \, \beta_{i_{k+1}} \tag{4.2}$$
$$\tag{4.3}$$

Décodeur. Les intervalles sont associés de manière bijective aux suites de caractères. Le nombre n_1 qui code le message complet appartient à un intervalle qui détermine le premier caractère de ce message. Pour retrourver la suite des caractères qui composent le message, le décodeur procède de la manière suivante :

$$n_1 = \text{le nombre sorti du codeur} \tag{4.4}$$
$$n_{k+1} = \frac{n_k - \alpha_{i_k}}{p_{i_k}} \tag{4.5}$$
$$\tag{4.6}$$

A chaque itération, le nombre n_{k+1} appartient à un intervalle qui détermine le caractère $k+1$.

Exemple de codeur/décodeur

On considère une source à $N = 3$ états notés x_1 ='A', x_2 ='M', x_3 ='I' de probabilités respectives $p_1 = 0.44$, $p_2 = 0.16$ et $p_3 = 0.4$.

On veut procéder au codage arithmétique de la séquence, de longueur $n = 3$, $x_{i_1}x_{i_2}x_{i_3}$='MAI' ($i_1 = 2$, $i_2 = 1$, $i_3 = 3$) issue de cette source.

Codage

Initialisation. $a_0 = 0$, $b_0 = 1$ et $w_0 = 1$.

La partition initiale de $[0; 1)$ est

$I_1 = [\alpha_1 = 0; \beta_1 = 0.44)$, $I_2 = [\alpha_2 = 0.44; \beta_2 = 0.6)$, $I_3 = [\alpha_3 = 0.6; \beta_3 = 1)$.

Codage du caractère numéro 1 **($k = 0$, $i_1 = 2$)**

$$\begin{aligned} a_1 = a_0 + w_0\,\alpha_{i_1} &= 0 + 1 \times 0.44 = 0.44 \\ b_1 = a_0 + w_0\,\beta_{i_1} &= 0 + 1 \times 0.6 = 0.6 \end{aligned}$$

L'intervalle qui code la séquence d'un caractère 'M' est $C_1 = [0.44; 0.6]$ de longueur $w_1 = 0.16$.

Codage du caractère numéro 2 **($k = 1$, $i_2 = 1$)**

$$\begin{aligned} a_2 = a_1 + w_1\,\alpha_{i_2} &= 0.44 + 0.16 \times 0 = 0.44 \\ b_2 = a_1 + w_1\,\beta_{i_2} &= 0.44 + 0.16 \times 0.44 = 0.5104 \end{aligned}$$

L'intervalle qui code la séquence 'MA' est $C_2 = [0.44; 0.5104]$ de longueur $w_2 = 0.0704$.

Codage du caractère numéro 3 **($k = 2$, $i_3 = 3$)**

$$\begin{aligned} a_3 = a_2 + w_2\,\alpha_{i_3} &= 0.44 + 0.0704 \times 0.6 = 0.4822 \\ b_3 = a_2 + w_2\,\beta_{i_3} &= 0.44 + 0.0704 \times 1 = 0.5104 \end{aligned}$$

L'intervalle qui code la séquence 'MAI' est $C_3 = [0.4822; 0.5104]$ de longueur $w_3 = 0.0282$.

Finalement, le codage de la séquence 'MAI' est tout nombre de l'intervalle $C_3 = [0.4822; 0.5104]$, prenons $n = 0.5$.

Décodage

Partant du nombre $n = 0.5$, les étapes du décodage sont :

Initialisation. $n_1 = 0.5$.

Décodage du caractère numéro 1 **($k = 0$).**

$n_1 = 0.5 \in I_2$: le premier caractère appartient à l'intervalle du 'M' ($i_1 = 2$) de probabilité $p_{i_1} = p_2 = 0.16$.

Décodage du caractère numéro 2 **($k = 1$).**

$$n_2 = \frac{n_1 - \alpha_{i_1}}{p_{i_1}} = (0.5 - 0.44)/0.16 = 0.375$$

$n_2 = 0.375 \in I_1$: le deuxième caractère appartient à l'intervalle du 'A' ($i_2 = 1$) de probabilité $p_{i_2} = 0.44$.

Décodage du caractère numéro 3 $(k = 2)$.

$$n_3 = \frac{n_2 - \alpha_{i_2}}{p_{i_2}} = (0.375 - 0)/0.44 = 0.85$$

$n_3 = 0.85 \in I_3$: le dernier caractère appartient à l'intervalle du 'I' $(i_3 = 3)$ de probabilité $p_{i_3} = 0.4$.

Remarque sur la mise en œuvre du codage arithmétique

En pratique, utiliser des nombre en virgule flottante peut poser problème : il est préférable de représenter l'intervalle initial sous la forme $0, 1, \cdots, N_{\max}$ et de ne manipuler que des entiers. Dans les deux cas, l'algorithme bute sur le caractère fini de la représentation et un nombre ne peut coder qu'une suite de longueur finie L qu'il est nécessaire de déterminer avant de procéder au codage par bloc de longueur L des messages.

4.4 Aspects pratiques du codage entropique

En pratique, plusieurs solutions sont possibles :

Loi connue a priori. Si le codeur et le décodeur disposent a priori du jeu de probabilité de la source et si les fréquences empiriques (les probabilités estimées à partir du fichier lui-même) coïncident avec ces probabilités a priori utilisées pour le codage et le décodage, la procédure de codage est optimale. En général bien sûr, cela n'est pas le cas et le déajustement entre la loi a priori et la loi empirique induit une sous optimalité dont le coût par caractère peut être évalué : il est donné par la divergence de Kullback entre les deux lois.

Loi inconnue a priori. Pour éviter ce problème de désajustement entre la loi empirique et celle qui est utilisée par le codeur et le décodeur, il est possible d'utiliser les fréquences empiriques pour le codage, mais alors, cette loi doit être transmise au décodeur en supplément du message compressé lui-même.

Pour choisir entre ces deux solutions, il faut comparer ces deux pénalités. Typiquement, pour des messages courts, le surcoût lié à la transmission au décodeur de la loi empirique ne compense en général pas le gain de codage alors que pour des messages longs la pénalité due au déajustement l'emporte et embarquer le jeu des probabilités empiriques avec le message devient une meilleure option.

4.5 Suites typiques

Ce paragraphe donne une idée intuitive de la raison pour laquelle la non maximalité de l'entropie d'une source simple permet de réduire l'ensemble des messages pour lesquels une compression est nécessaire.

La possibilité de réécrire de manière plus compacte les messages délivrés par une source simple (c'est-à-dire qui génère des v.a. i.i.d.) peut être comprise intuitivement grâce à la notion de suite typique : lorsque l'entropie d'une source n'est pas maximale seule une faible partie de l'ensemble des suites possibles se réalisent avec une probabilité significative. Dans la limite des longues suites, l'ensemble des messages peut ainsi être divisé en deux sous-ensembles : l'un comprend les suites typiques qui apparaissent toutes avec la même probabilité et l'autre des suites dont la probabilité d'occurrence est négligeable. Il est ainsi possible de n'associer des étiquettes (mots de code) qu'aux seules suites typiques, leur nombre étant très inférieur au cardinal de l'ensemble de toutes les suites possibles, l'écriture des messages est plus compacte que celle qui résulte d'un étiquetage indifférencié de toutes les suites possibles.

Une suite s_n de n symboles est un ensemble ordonné de n réalisations indépendantes d'une v.a. X pouvant prendre N états $x_1, \cdots, x_N$ avec les probabilités $p_i = p(x_i)$.

Soit $\varepsilon > 0$ et k tels que $1/k^2 < \varepsilon/N$. Une suite est dite typique si elle vérifie la condition suivante pour tous ses symboles :

$$\frac{|f_i(s_n) - np_i|}{\sqrt{np_i(1-p_i)}} < k, \forall i \in \{1, \cdots, N\}$$

$f_i(s_n)$ représente le nombre d'occurrences du symbole i dans la suite s_n de longueur n, c'est une variable aléatoire binomiale de moyenne np_i et d'écart type $\sqrt{np_i(1-p_i)}$. Une suite est donc dite typique lorsque la fréquence attendue np_i de chacun des symboles i est proche de la fréquence empirique $f_i(s_n)$ effectivement observée dans la suite s_n. Ici, le terme proche signifie simplement que l'écart est de l'ordre de $\sqrt{n}$ alors que le nombre d'éléments de la suite vaut n.

Point 1 : L'ensemble des suites non-typiques est asymptotiquement négligeable.

$$P\left(s_n \notin T\right) = P\left(\frac{\left|f_i\left(s_n\right) - np_i\right|}{\sqrt{np_i\left(1-p_i\right)}} > k \text{ pour au moins un des } x_i\right)$$

En majorant la probabilité de l'union des évènements par la somme de leurs probabilités, on peut écrire :

$$P\left(s_n \notin T\right) \leqslant \sum_{i=1}^{N} P\left(\frac{\left|f_i\left(s_n\right) - np_i\right|}{\sqrt{np_i\left(1-p_i\right)}} > k\right)$$

D'après l'inégalité de Tchebychev, la probabilité pour qu'une v.a. s'écarte de sa moyenne de plus de k fois son écart type est inférieure à $1/k^2$, ainsi :

$$P\left(s_n \notin T\right) \leqslant \sum_{i=1}^{N} \frac{1}{k^2} = \frac{N}{k^2} < \varepsilon$$

La probabilité pour que la suite ne soit pas typique est donc négligeable.

Point 2 : chaque suite typique a une probabilité voisine de 2^{-nH}. Par définition, pour chaque symbole i d'une suite typique s_n, on a :

$$np_i - k\sqrt{np_i\left(1-p_i\right)} \leqslant f_i(s_n) \leqslant np_i + k\sqrt{np_i\left(1-p_i\right)}$$

Posons $A = -k\sum_{i=1}^{N} \log\left(p_i\right)\sqrt{p_i\left(1-p_i\right)}$. En multipliant l'inégalité précédente par $-\log_2 p_i > 0$ et en sommant sur l'ensemble des N symboles possibles :

$$nH - A\sqrt{n} \leqslant -\sum_{i=1}^{N} f_i(s_n)\log(p_i) \leqslant nH + A\sqrt{n}$$

En remarquant maintenant que la probabilité de la suite s_n vaut $p(s_n) = p_1^{f1(s_n)} p_2^{f_2(s_n)} \cdots p_N^{f_N(s_n)}$, on a :

$$-\log P(s_n) = -\sum_{i=1}^{N} f_i(s_n)\log(p_i)$$

Finalement, la probabilité $p(s_n)$ peut être encadrée comme suit :

$$2^{-nH-A\sqrt{n}} \leqslant P(s_n) \leqslant 2^{-nH+A\sqrt{n}}$$

Pour n grand, chaque suite typique s_n possède une probabilité voisine de 2^{-nH} .

Point 3 : Le nombre de suites typiques est de l'ordre de 2^{nH} Notons ν le nombre de suites typiques de longueur n . L'inégalité précédente étant valable pour toute suite typique, une sommation sur l'ensemble des suites typiques permet d'écrire : $\nu 2^{-nH-A\sqrt{n}} \leqslant p(s_n \in T) \leqslant \nu 2^{-nH+A\sqrt{n}}$. Or $1-\varepsilon \leqslant p(s_n \in T) \leqslant 1$. De ces deux inégalités, on tire l'encadrement :

$$(1-\varepsilon)2^{nH-A\sqrt{n}} \leqslant \nu \leqslant 2^{nH+A\sqrt{n}}$$

Pour n grand, le nombre de suites typiques est donc de l'ordre de 2^{nH}.

En résumé, pour des messages longs (n grand), on compte environ 2^{nH} suites typiques qui apparaissent toutes avec une même probabilité (voisine de 2^{-nH}). Les autres suites n'apparaissent pas avec une probabilité suffisamment élevée pour que leur compression soit nécessaire.

Lien entre codage bloc et suites typiques. En codant des extensions d'ordre s de la source, avec s grand, le nombre de suites typiques de longueur s est de l'ordre de $2^{sH(X)} = D^{sH_{D(X)}}$. La probabilité pour qu'une suite soit typique étant de l'ordre de un, on peut ne coder que les suites typiques. Avec un alphabet de D caractères, il faut que la longueur moyenne des mots du code soit égale à $sH_{D(X)}$. Autrement dit, la longueur moyenne des mots par caractère source tend vers $H_D(X)$ lorsque s tend vers l'infini.

Vue d'ensemble d'une chaîne classique de l'information

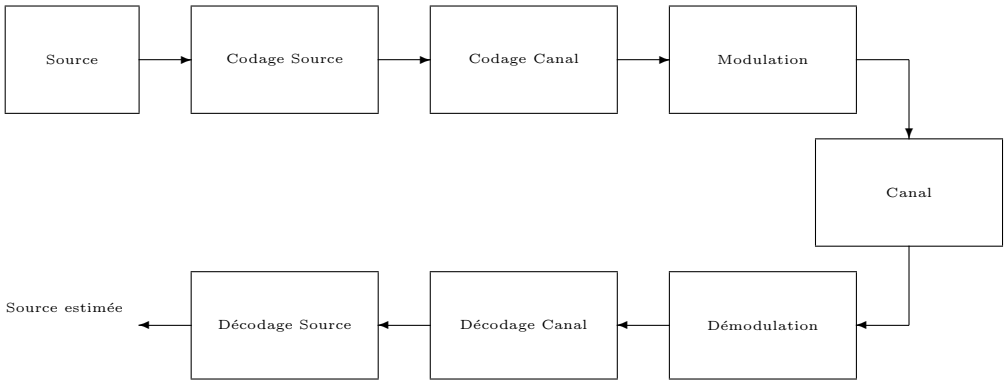


Figure 4.3: Structure générique d'une chaîne de transmission de l'information

4.6 Éléments d'une chaîne de l'information

Une chaîne de transmission de l'information vise à transmettre de manière rapide et fiable un message d'un expéditeur vers un destinataire. Son schéma général est donné par la figure (4.3).

Les différents éléments qui composent cette chaîne sont les suivants :

Le canal. Il peut s'agir

- d'un medium de transmission tel qu'une ligne de téléphone, un câble coaxial, une fibre optique ou une liaison radio (téléphone mobile, wifi, bluetooth, wimax, ...)
- d'un système physique de stockage de l'information, par exemple un disque magnétique ou optique, de la mémoire flash, de la RAM ou même d'un simple papier sur lequel est imprimé un message (code barre, code QR, filigrane).

Une caractéristique essentielle du canal réside dans sa non fiabilité, c'est-à-dire dans le fait que des perturbations de nature aléatoire affectent les messages qui y transitent. Le type de perturbation dépend du canal : bruit d'origine électronique, interférences entre utilisateurs lors d'une transmission radio, poussières, rayures ou taches sur un support de stockage optique.

La source. La source délivre un message informatif. On distingue classiquement deux types de sources :

1. les sources numériques telles qu'un fichier ou une image numérique.
2. les sources analogiques telles que la voix ou la musique. Les sources analogiques sont pratiquement toujours converties en sources numériques plus faciles à manier techniquement, le passage du monde analogique au monde numérique est appelé *numérisation* et se compose le plus souvent de deux étapes nommées *échantillonnage* et *quantification*.

La source numérique (obtenue directement ou par numérisation) peut être modélisée de manière probabiliste, le modèle le plus simple est celui d'une suite de variables aléatoires indépendantes identiquement distribuées (i.i.d.), c'est la source simple.

Le codeur de source. Le rôle du codeur de source est de représenter le message issu de la source de façon aussi concise que possible. On distingue deux types de codeurs de source :

1. les codeurs avec perte : les codeurs de ce type assurent des taux de compression qui peuvent être très élevés, le prix à payer pour atteindre une forte compression est que le codage n'est pas réversible, c'est-à-dire qu'il n'est pas possible de revenir de manière exacte du message compressé à l'original. Le codage avec perte est intéressant pour des domaines dans lesquels une certaine dégradation de la qualité est acceptable. Les applications communes concernent la parole, la musique, l'image et la vidéo et des codeurs classiques pour ces cas sont typiquement le CELP (parole), le MP3 (musique), le JPEG (image) et le MPEG (vidéo).
2. les codeurs sans perte : les codeurs de ce type assurent des taux de compression plus modestes que les codeurs avec perte mais le codage est réversible. Cette propriété est indispensable dans certain cas, par exemple lorsque le message est un fichier exécutable.

Le codeur canal. Alors que le codeur de source a à charge d'éliminer, ou tout au moins de réduire, la redondance "incontrôlée" du message initial, le codeur de canal introduit de la redondance contrôlée en vue de protéger le message lors de sa transmission sur un canal non fiable. Fondamentalement, le principe de tout codeur canal consiste à répéter l'information transmise afin de permettre au récepteur de détecter voire de corriger les erreurs introduites lors du passage au travers du canal.

Si le fait que le taux d'erreur tende vers zéro lorsque le nombre de répétitions tend vers l'infini (mais au prix d'un débit qui tend lui aussi vers zéro) est assez intuitif (loi des grands nombres), l'un des résultats les plus remarquables de la théorie de l'information est qu'il est également possible d'assurer un taux d'erreur aussi faible que souhaité sans réduire à zéro le débit pourvu que celui-ci soit inférieur à une certaine limite liée aux caractéristiques du canal.

Le modulateur. Le rôle du modulateur consiste à adapter le message numérique à la nature physique du canal. Par exemple en transformant une suite binaire en une séquence d'allumages et d'extinction d'une source lumineuse ou en une tension variant au fil du temps.

Le démodulateur. Dual du modulateur assurant idéalement (pour un canal parfait) la restitution de la séquence numérique en entrée du modulateur à partir de l'ensemble des observations effectuées en sortie du canal.

Le décodeur canal. Dual du codeur de canal assurant la détection ou la correction, dans la mesure du possible, des erreurs introduites par le canal. Idéalement, le décodeur corrige toutes les erreurs produites par un canal non fiable dans certaines limites théoriques liées au niveau de perturbation du canal, à la quantité d'information produite par la source et à la taille du message transmis.

Le décodeur source. Dual du codeur de source visant à la restitution du message en entrée du codeur de source de manière exacte (codeur sans perte) ou approchée (codeur avec perte). Le codeur de source assure une compression, le décodeur de source une décompression.

4.7 Quelques mots sur la modulation.

L'entrée d'un canal physique (tel qu'un coaxial ou une ligne téléphonique par exemple) est un signal, c'est-à-dire une quantité physique (lumière, électricité, onde sonore, ...) porteuse d'information. Les signaux sont typiquement des fonctions du temps mais ils peuvent également dépendre de l'espace (image, code barre) ou de tout autre paramètre. Dans ce paragraphe, le signal est une fonction du temps.

Appelons T la durée de la transmission, combien de fonctions orthogonales est-il possible de construire sur cette durée T ? Il est clair que la réponse est une infinité, par exemple la base de Fourier

$$\left\{\exp\left[i2\pi\nu\frac{n}{T}t\right]\right\}_{n\in\mathbb{Z}}$$

pour le produit scalaire

$$\langle f,g\rangle = \frac{1}{T}\int_0^T f(t)g^*(t)dt$$

Les canaux physiques sont de bande passante B limitée, cela signifie que seules les fréquences $\nu \in [0,B]$ passent au travers du canal. Ainsi, le nombre des fonctions de base qui passent est $\frac{B}{1/T} = BT$ et pour transmettre un ensemble de BT valeurs $\{c_n\}_{n=1\cdots BT}$, on transmet la fonction

$$f(t) = \sum_{n=1}^{BT} c_n \exp\left[i2\pi\nu\frac{n}{T}t\right]$$

C'est la modulation ; transformation d'un ensemble discret de valeurs en une fonction.

Un modèle de canal élémentaire est tel que la sortie $s(t)$ du canal est une version bruitée par un bruit $b(t)$ de son entrée :

$$s(t) = f(t) + b(t)$$

En calculant le produit scalaire de $s(t)$ avec chacune des fonctions $\exp\left[i2\pi\nu\frac{n}{T}t\right]$, on obtient BT valeurs :

$$\begin{array}{rcl} s_1 & = & c_1 + b_1 \\ \vdots & \vdots & \vdots \\ s_N & = & c_N + b_N \end{array}$$

avec $b_n = \left\langle b(t), \exp\left[i2\pi\nu\frac{n}{T}t\right]\right\rangle$. C'est la démodulation, opération duale de la modulation qui permet de passer du monde continu des fonctions au monde discret des suites.

En l'absence de bruit, le couple (modulateur, démodulateur) est transparent : la sortie du démodulateur est égale à l'entrée du modulateur.

Les opérations de modulation/démodulation (pratiquement groupées au sein d'un mo-dem) transforment le canal physique en BT canaux élémentaires à entrée scalaire et sortie scalaire.

Modulation et démodulation réalisent le lien entre les modèles de canaux classiques en théorie de l'information et le monde réel.

Chapter 5

Canal et Capacité

Contents

La source en entrée du canal génère des symboles, elle possède une certaine entropie et distille une certaine quantité d'information.

Lors de son passage au travers du canal, l'information est affectée par des perturbations et la question est de savoir quelle part de l'information transmise peut être récupérée en sortie du canal. La quantité d'information qui passe dans le canal peut être vue comme la différence entre l'incertitude quant à la source avant observation de la sortie du canal et l'incertitude conditionnelle à cette observation. Cette différence dépend de la loi d'entrée du canal, une maximisation sur cette loi d'entrée conduit à la notion de capacité. L'existence de cette notion théorique de capacité est essentielle du fait du deuxième théorème de Shannon qui montre qu'une transmission peut être fiable (par un moyen appelé codage) à condition que l'entropie de la source soit inférieure à la capacité du canal.

5.1 Notion de canal

Pratiquement un "canal de transmission" est par exemple un milieu physique au travers duquel il est possible de faire passer de l'information. La démarche adoptée ici ne se préoccupe en aucun cas de l'origine physique des propriétés du canal, elle se contente d'en donner une description purement probabiliste. Le canal est alors considéré comme un système probabiliste qui accepte des symboles porteurs d'information en entrée et restitue en sortie d'autres symboles. Les alphabets d'entrée et de sortie sont en général différents.

5.1.1 Canal discret sans mémoire et invariant

Notons $\{x_1, \cdots, x_N\}$ les N lettres de l'alphabet d'entrée du canal et $\{y_1, \cdots, y_M\}$ les M lettres de son alphabet de sortie.

En général, la séquence $y_{j_1}, \cdots, y_{j_n}$ observée en sortie d'un canal discret dépend de son entrée $x_{i_1}, \cdots, x_{i_n}$ et d'un état interne du canal. Du point de vue probabiliste, cela se traduit par le fait que la loi de probabilité entrée-sortie est de la forme :

$$P_n\left(y_{j_1}, \cdots, y_{j_n}; x_{i_1}, \cdots, x_{i_n};\ \text{état}\right)$$

où les y_i appartiennent à l'alphabet de sortie du canal et les x_i à l'alphabet d'entrée. Cette relation signifie que le symbole présent à un instant donné en sortie du canal dépend de l'ensemble de symboles présents à l'entrée et de l'état propre du canal avant l'application de la première entrée x_1. Le canal est dit sans mémoire lorsque cette relation entrée-sortie peut être réduite à :

$$\begin{aligned} P_n\left(y_{j_1}, \cdots, y_{j_n}; x_{i_1}, \cdots, x_{i_n};\ \text{état}\right) &= P_n\left(y_{j_1}, \cdots, y_{j_n}; x_{i_1}, \cdots, x_{i_n}\right) \\ &= P_1\left(y_{j_1}|x_{i_1}\right) \cdots P_n(y_{j_n}|x_{i_n}) \end{aligned}$$

La première égalité signifie que le canal ne possède pas d'état interne et la seconde que le symbole observé en un indice donné en sortie ne dépend que de l'entrée au même indice. Dans le cas d'un canal sans mémoire, le lien probabiliste entre l'entrée et la sortie est complètement décrit par la donnée des jeux de probabilités de transition $P_k(y_{j_k}|x_{i_k})$, $k = 1 \cdots n$. Si les caractéristiques du canal sont invariantes $P_k(y_{j_k}|x_{i_k})$ ne dépend pas de k et ce jeu de probabilités est représentable sous forme matricielle : la matrice de transition $N \times M$ d'un canal discret sans mémoire est alors définie par :

$$\mathbf{\Pi} = [\Pi_{ij}] = [p(y_j|x_i)] \quad \begin{matrix} i \in \{1 \cdots N\} \\ j \in \{1 \cdots M\} \end{matrix}$$

Remarques :

- Chaque ligne de la matrice de transition $\mathbf{\Pi}$ contient un jeu de probabilités, d'où $\sum_{j=1}^{M} p\left(y_j|x_i\right) = 1, \forall i \in \{1, \cdots, N\}$. Ce n'est pas le cas pour les colonnes.
- La loi de probabilité de la sortie Y du canal s'obtient simplement à partir de celle de l'entrée X et de la matrice de transition[1] :

  $$\mathcal{P}_Y = \mathbf{\Pi}^T \mathcal{P}_X$$

  avec

  $$\begin{aligned} \mathcal{P}_Y^T &= [p(y_1), \cdots, p(y_M)] \\ \mathcal{P}_X^T &= [p(x_1), \cdots, p(x_N)] \end{aligned}$$

Quelques structures particulières utiles :

Canal uniforme par rapport à l'entrée. Pour un tel canal, les symboles sont tous affectés de la même manière par les erreurs : les lignes sont identiques à une permutation près. Une conséquence importante de cette uniformité en entrée est que l'entropie conditionnelle ne dépend pas de la loi de l'entrée $H(Y|X) = H(Y|X = x_i), \forall i \in \{1, \cdots, N\}$.

Canal uniforme par rapport à la sortie. Les colonnes sont identiques à une permutation près. Une conséquence importante de cette uniformité en sortie est qu'une loi uniforme en entrée induit une loi uniforme en sortie.

Canal symétrique. C'est un canal uniforme en entrée et en sortie avec $N = M$.

5.1.2 Canaux élémentaires

Deux canaux très simples jouent un rôle fondamental : le canal binaire symétrique et le canal à bruit additif gaussien.

[1] Cette relation est la forme vectorielle de $p(y_j) = \sum_{i=1}^{N} p\left(y_j|x_i\right) p\left(x_i\right)$

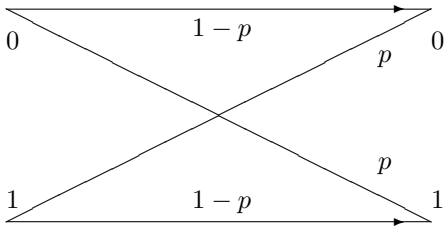


Figure 5.1: Canal binaire symétrique.

Le Canal Binaire Symétrique (CBS)

Le canal binaire symétrique (cf. figure 5.1) est à entrées binaires (notées par exemple 0 et 1, mais cela n'a pas d'importance, ce ne sont que des étiquettes) et à sorties binaires (également notées 0 et 1) tel que :

- Le canal est sans mémoire : la sortie en k dépend uniquement de l'entrée en k.
- Binaire : 2 entrées possibles $(0,\ 1)$ et 2 sorties possibles $(0,\ 1)$.
- Symétrique : les entrées 0 et 1 sont affectées de manière égale par les erreurs (probabilité d'erreur p). Les probabilités de transition, et donc les performances du canal, sont complètement déterminées par un seul paramètre p :

$$\begin{aligned} p &= Pr(y_k = 1|C_k = 0) = Pr(y_k = 0|C_k = 1) \\ 1-p &= Pr(y_k = |C_k = 1) = Pr(y_k = 0|C_k = 0) \end{aligned}$$

La matrice de transition $\mathbf{\Pi} = [\Pi_{ij}] = [p(y_j|x_i)]_{i,j\in\{1;2\}}$ d'un canal binaire symétrique est une matrice 2×2 donnée par :

$$\mathbf{\Pi} = \begin{bmatrix} 1-p & p \\ p & 1-p \end{bmatrix}$$

Vraisemblance des observations en sortie de canal. Si l'entrée est un message constitué d'une suite de n valeurs binaires $C_1, \cdots, C_n$ que nous groupons dans un vecteur $\boldsymbol{C}$, la distance de Hamming entre le vecteur $\boldsymbol{y} = [y_1, \cdots, y_n]^T$ composé des n valeurs binaires observées en sortie du canal et le vecteur $\boldsymbol{C}$ composé des valeurs binaires placées en entrée est le nombre de positions qui diffèrent entre ces deux vecteurs

$$d_H(\boldsymbol{y}, \boldsymbol{C}) = |\{j|0 \leq j \leq n, y_j \neq C_j\}|$$

La vraisemblance des observations $\boldsymbol{y}$ s'écrit :

$$p(\boldsymbol{y}|\boldsymbol{C}_{rs}) = (1-p)^n \left(\frac{p}{1-p}\right)^{d_H(\boldsymbol{y},\boldsymbol{C}_{rs})}$$

d'où

$$\log p(\boldsymbol{y}|\boldsymbol{C}_{rs}) = d_H(\boldsymbol{y}, \boldsymbol{C}_{rs}) \log\left(\frac{p}{1-p}\right) + n \log(1-p)$$

$n \log(1-p)$ est une constante et $\log\left(\frac{p}{1-p}\right) < 0$ (car $0 < p < 1/2$), ainsi maximiser la vraisemblance revient à minimiser $d_H(\boldsymbol{y}, \boldsymbol{C}_{rs})$.

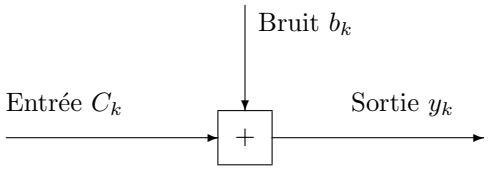


Figure 5.2: Canal à bruit additif blanc gaussien. A chaque utilisation canal, la sortie y_k est la somme de l'entrée C_k et d'une perturbation b_k distribuée selon une loi normale de moyenne nulle et de variance σ^2. La suite des v.a. b_k est i.i.d. : le canal est sans mémoire.

Canal à bruit additif gaussien

Le canal gaussien (cf. figure (5.2)) est à entrée réelle et à sortie réelle tel que :

- Le canal est sans mémoire : la sortie dépend uniquement de l'entrée actuelle.
- La sortie est la superposition de l'entrée et d'une v.a. gaussienne centrée de variance σ^2 (le bruit gaussien) de densité de probabilité
$$p\left(x\right) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{x^2}{2\sigma^2}\right)$$
- Les performances du canal sont complètement déterminées par un seul paramètre σ^2.

Bien que le canal gaussien ne soit pas à état fini en entrée ni en sortie, il est présenté ici car il joue un rôle important en pratique lorsque l'on s'intéresse à l'aspect physique d'un stockage ou d'une transmission de l'information.

Vraisemblance des observations en sortie de canal. Si une valeur C_k est placée en entrée d'un canal gaussien, sa sortie y_k suit une loi normale de variance σ^2 centrée en C_k que nous pouvons noter $p(y_k|C_k)$ ou $p_{C_k}(y_k)$:
$$p(y_k|C_k) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{|y_k - C_k|^2}{2\sigma^2}\right)$$
Si maintenant, l'entrée est un message constitué d'une suite de n valeurs $C_1, \cdots, C_n$ que nous groupons dans un vecteur $\boldsymbol{C} = [C_1, \cdots, C_n]^T$.

La distance euclidienne entre le vecteur $\boldsymbol{y} = [y_1, \cdots, y_n]^T$ composé des n valeurs observées en sortie du canal et le vecteur $\boldsymbol{C}$ composé des valeurs placées en entrée est la somme des carrés des erreurs entre ces deux vecteurs
$$d_E^2\left(\boldsymbol{y}, \boldsymbol{C}\right) = \|\boldsymbol{y} - \boldsymbol{C}\|^2 = \sum_{j=1}^{n} |y_j - C_j|^2$$
Lorsque la suite des perturbations b_k est une suite de v.a. indépendantes de même loi, la vraisemblance des observations $\boldsymbol{y}$ s'écrit :
$$\begin{aligned} p(\boldsymbol{y}|\boldsymbol{C}) &= \prod_{j=1}^{n} \left[\frac{1}{\sigma\sqrt{2\pi}}\right] \exp\left(-\frac{|y_j - C_j|^2}{2\sigma^2}\right) \\ &= \left[\frac{1}{\sigma\sqrt{2\pi}}\right]^n \exp\left(-\frac{\|\boldsymbol{y} - \boldsymbol{C}\|^2}{2\sigma^2}\right) \\ &\propto \exp\left(-\frac{d_E^2\left(\boldsymbol{y}, \boldsymbol{C}\right)}{2\sigma^2}\right) \end{aligned}$$

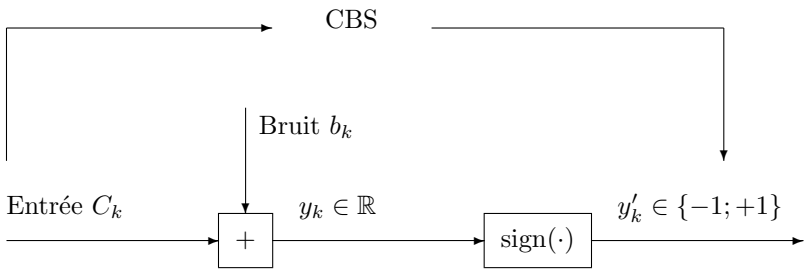


Figure 5.3: Un canal à bruit additif blanc gaussien à entrées binaires avec décisions binaires en sortie est un canal binaire symétrique (CBS) de paramètre $p = Q\left(1/\sigma\right)$.

La fonction $\log$ étant monotone, maximiser le logarithme de la vraisemblance équivaut à maximiser la vraisemblance elle-même et conduit à des calculs plus simples :

$$\log p(\boldsymbol{y}|\boldsymbol{C}) = -n \log\left[\sigma\sqrt{2\pi}\right] - \frac{d_E^2\left(\boldsymbol{y},\boldsymbol{C}\right)}{2\sigma^2}$$

Ainsi, maximiser la vraisemblance à variance connue est équivalent à minimiser la distance euclidienne et il est possible d'estimer, au sens du maximum de vraisemblance (MV), l'entrée $\boldsymbol{C}$ à partir des sorties $\boldsymbol{y}$ en minimisant une distance Euclidienne.

La canal binaire vu comme approximation du canal gaussien

Supposons un canal gaussien à entrée binaire $C_k = \pm 1$. Lorsque $C_k = +1$, la sortie du canal suit une loi normale de moyenne $+1$; lorsque $C_k = -1$, une loi normale de moyenne -1.

La règle de décision optimale au sens du MV consiste à prendre le signe de la sortie du canal gaussien et ainsi à transformer celle-ci en une valeur binaire. On créé de cette manière un canal binaire symétrique entre les entrées binaires du canal gaussien et les décisions binaires prises en sortie (cf. figure 5.3).

En ce sens, le canal binaire symétrique est une approximation du canal gaussien avec :

$$p = \frac{1}{\sigma\sqrt{2\pi}} \int_1^{+\infty} \exp\left(-\frac{x^2}{2\sigma^2}\right) dx = Q\left(\frac{1}{\sigma}\right)$$

avec

$$Q(x) = (2\pi)^{-1/2} \int_x^{+\infty} \exp\left(-t^2/2\right) dt$$

Cette approximation fait perdre la fiabilité des décisions : pour une observation positive proche de 0, les hypothèses ± 1 sont presque aussi vraisemblables l'une que l'autre alors que pour une observation positive très supérieure à 0 l'hypothèse $+1$ est beaucoup plus vraisemblable que -1. Une fois la décision prise, cette information est perdue.

5.2 Capacité

Les hypothèses faites dans la suite sont les suivantes :

- l'entrée du canal est une suite de réalisations indépendantes d'une v.a. X. Cette hypothèse est naturelle puisque l'un des buts du codage de source est précisément de rendre les caractères après compression indépendants les uns des autres.
- le canal est invariant et sans mémoire de sorte que, pour une entrée X, sa sortie Y est liée seulement à X.

5.2.1 Définition de la capacité

L'observation de la sortie Y diminue l'incertitude sur l'entrée X ; elle apporte de l'information. Avant observation de Y, l'incertitude *a priori* sur l'entrée X est $H(X)$. Après observation de Y, elle est plus faible et vaut $H(X|Y) \leq H(X)$. La réduction d'incertitude est la quantité d'information apportée par Y sur X, elle vaut :

$$I(X;Y) = \overbrace{H(X)}^{\text{Incertitude } a\ priori} - \overbrace{H(X|Y)}^{\text{Incertitude } a\ posteriori} \geq 0$$

Étant donné que le conditionnement réduit l'incertitude ($0 \leq H(X|Y) \leq H(X)$), on a $0 \leq I(X;Y) \leq H(X)$.

Cette information mutuelle est positive ou nulle, elle dépend de la loi de X et de la nature du canal. Puisque qu'il est impossible de modifier la nature physique du canal, la maximisation de l'information qui traverse le canal se fait en choisissant au mieux la loi $\mathcal{P}_X$ de l'entrée, d'où la définition de la capacité C par l'information maximale qui peut passer au travers de ce canal :

$$C = \max_{\mathcal{P}_X} I(X;Y)$$

L'information mutuelle est une fonction convexe de la loi de l'entrée (exercice, le montrer), ainsi la recherche du maximum de l'information mutuelle se réduit à celle d'un extremum.

Signification des différents termes impliqués dans la capacité :

$H(X)$ Entropie de la source.

$H(X|Y)$ Incertitude résiduelle sur X sachant Y (Cette quantité est parfois appelée équivoque.). De l'incertitude subsiste du fait que le canal est bruité.

$I(X;Y)$ L'information qui passe entre X et Y ; c'est-à-dire, l'information mutuelle étant symétrique, l'information partagée par les v.a. X et Y.

En général, le calcul de la capacité d'un canal est difficile. Le paragraphe suivant traite d'un cas simple mais réaliste : le canal symétrique.

5.2.2 Capacité d'un canal symétrique.

Le calcul de la capacité d'un canal symétrique peut être mené à bien sans difficulté. Rappelons que l'information mutuelle est symétrique $I(X;Y) = I(Y;X) = H(Y) - H(Y|X)$. Le calcul se décompose en plusieurs étapes :

1. Monter que l'entropie conditionnelle $H(Y|X)$ est indépendante de la loi d'entrée. Ce résultat permet de ramener la maximisation de l'information mutuelle à celle de $H(Y)$.

   Preuve. On a
   $$H(Y|X) = \sum_{k=1}^{N} p(x_k) H(Y|X = x_k)$$
   avec
   $$H(Y|X = x_k) = -\sum_{j=1}^{M} p(y_j|x_k) \log p(y_j|x_k)$$

Les quantités $p(y_j|x_k)$ représentent les éléments de la k-ième ligne de la matrice de transition du canal. Le canal étant symétrique, *a fortiori* uniforme en entrée, le jeu de probabilités est le même sur toutes les lignes et l'entropie $H(Y|X=x_k)$ est indépendante de k, d'où

$$\begin{aligned} H(Y|X) &= \sum_{k=1}^{N} p(x_k)\, H(Y|X=x_k) \\ &= H(Y|X=x_i)\left[\sum_{k=1}^{N} p(x_k)\right],\ \forall i = 1\cdots N \\ &= H(Y|X=x_i),\ \forall i = 1\cdots N \end{aligned}$$

Finalement l'entropie conditionnelle

$$H(Y|X) = -\sum_{j=1}^{M} p(y_j|x_i)\log p(y_j|x_i),\ \forall i = 1\cdots N$$

ne dépend que des probabilités de transition du canal $p(y_j|x_i)$, elle est indépendante de la loi d'entrée. □

2. Montrer que $H(Y)$ est maximum lorsque la loi d'entrée est uniforme.

   Preuve. $H(Y)$ est maximale lorsque Y suit une loi uniforme. Il suffit donc de déterminer la loi de X qui rend Y uniforme. Montrons que la loi uniforme pour X est solution. Si X est uniforme, on a $p(x_i) = 1/N$ pour tout i. La probabilité du symbole y_j en sortie est donnée par

   $$p(y_j) = \sum_{i=1}^{N} p(x_i, y_j) = \sum_{i=1}^{N} p(x_i)p(y_j|x_i) = \frac{1}{N}\sum_{i=1}^{N} p(y_j|x_i)$$

   La somme $\sum_{i=1}^{N} p(y_j|x_i)$ représente la somme des éléments de la colonne j : elle est indépendante de j car le canal est symétrique, *a fortiori* uniforme en sortie. Ainsi, une entrée de loi uniforme maximise l'information mutuelle et permet d'atteindre la capacité. □

3. Calculer la capacité

   Valeur de la capacité du canal symétrique. Il suffit d'évaluer l'information mutuelle pour une loi d'entrée uniforme. On a

   $$C = H(Y) - H(Y|X) = \log(M) + \sum_{j=1}^{M} p(y_j|x_i)\log p(y_j|x_i)$$

   Cette formule est valable pour un canal doublement uniforme (en entrée et en sortie). Si le canal est symétrique on a aussi $N = M$. □

Cas du canal binaire symétrique. Pour un CBS, la connaissance de la seule quantité scalaire p détermine complètement la matrice de transition du canal et sa capacité ne dépend que cette probabilité d'erreur p :

$$C(p) = 1 + (1-p)\log(1-p) + p\log(p)$$

En notant $H(p, 1-p)$ l'entropie d'une loi binominale, on a $C = 1 - H(p, 1-p)$.

Cas sans bruit. Pour p nul ou égal à un, la capacité est maximale, le canal est sans bruit.

Un canal sans bruit est idéal pour la transmission ou le stockage de l'information.

Canal de capacité nulle. Pour $p = 1/2$, la capacité est nulle. En effet, un symbole en sortie du canal peut provenir avec la même probabilité de l'un ou de l'autre des symboles d'entrée. Autrement dit, l'observation de la sortie ne renseigne en rien l'entrée : l'information apportée par cette observation est nulle.

Un canal de capacité nulle est idéal pour la dissimulation de l'information.

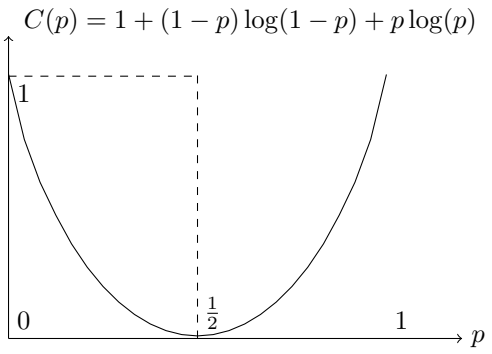


Figure 5.4: Capacité d'un CBS.

La capacité d'un canal binaire symétrique (CBS) est maximale en $p = 0$ et $p = 1$ (canal sans bruit), elle est nulle en $p = 1/2$ (canal de capacité nulle). Le pire cas (pour la transmission) est donc $p = 1/2$ et non pas 1, valeur pour laquelle il suffit d'inverser chacune des sorties pour restaurer parfaitement l'entrée.

5.3 Second théorème de Shannon

L'importance de la capacité tient à un résultat essentiel dû à C. Shannon qui affirme que, pour peu que l'entropie de la source placée en entrée du canal soit strictement inférieure à la capacité de celui-ci, il est possible, sous certaines conditions, de récupérer parfaitement l'entrée à partir de la sortie. Ce résultat peu intuitif permet de construire des systèmes de transmission ou de stockage fiables sur des canaux qui ne le sont fondamentalement pas.

Le moyen pratique qui permet d'atteindre cet objectif de fiabilité est le codage canal (aussi appelé codage détecteur et correcteur d'erreurs). Contrairement au codage de source qui vise à décrire les messages délivrés par la source de la manière la plus concise possible, le codage de canal augmente la longueur des messages par adjonction de redondance en vue de permettre une correction des erreurs dues aux imperfections du canal lors de l'estimation de l'entrée du canal à partir de sa sortie.

5.3.1 Code à répétition

Commençons par un exemple très simple destiné à illustrer le caractère étonnant du deuxième théorème de Shannon.

Soit un canal binaire symétrique, de probabilité d'erreur p, à entrées et sorties dans $\{0, 1\}$. On cherche à réduire cette probabilité d'erreur en ajoutant au message une certaine redondance. La méthode la plus simple consiste à répéter plusieurs fois chaque symbole en entrée du canal pour procéder ensuite à une décision majoritaire en sortie.

Supposons que le même symbole x (valant 0 ou 1) soit émis $n = 2s + 1$ fois (s entier), la décision peut procéder de la règle suivante :

- si le nombre de x reçu est supérieur ou égal à $s + 1$, on décide que x a été émis,
- sinon, on décide $1 - x$.

Illustration pour $n = 3$. La figure (5.5) illustre les mots d'un code à répétition pour $n = 3$. Les mots du code (000) et (111) sont représentés par les grandes sphères.

- Si l'un des mots du code est reçu, on décide que c'est bien ce mot qui est présent en entrée. Cette décision, bien que non certaine, est très fiable lorsque les erreurs qui affectent les valeurs transmises sont indépendantes. En effet, si (000) est en entrée et que, par exemple, la probabilité d'erreur vaut $p = 10^{-2}$, la probabilité d'observer ce même mot (000) en sortie vaut $(1 - 10^{-2})^3 \approx 0.97$ alors que celle d'observer le mot (111) vaut $(10^{-2})^3 = 10^{-6}$.

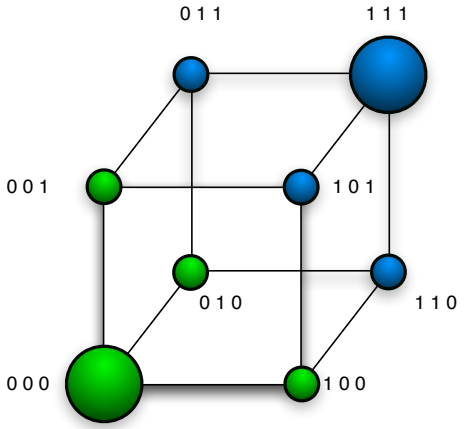


Figure 5.5: k=1 bit d'information, 2 mots de code (de longueur n=3): (000) (111).

- De même lorsqu'une séquence autre que (000) ou (111) est observée, il est plus probable qu'elle provienne de celui des deux mots du code qui se trouve à distance de Hamming minimale et décider qu'elle provient effectivement de ce mot est la règle de décodage qui minimise la probabilité d'erreur.

  Par exemple, si l'on observe 001, la probabilité pour que cette observation provienne du mot d'entrée 000 (1 erreur en position 3) vaut $(1-10^{-2})^2 10^{-2} \approx 10^{-2}$ alors que la probabilité pour qu'elle provienne du mot 111 (2 erreurs aux positions 1 et 2) vaut $(1-10^{-2})\left(10^{-2}\right)^2 \approx 10^{-4}$. L'hypothèse '$000$ en entrée' est environ cent fois plus vraisemblable que l'hypothèse '111 en entrée'.

Probabilité d'erreur par mot. Le canal étant symétrique, la probabilité d'erreur est la même pour le 0 et pour le 1. Ainsi, il est possible de calculer la probabilité d'erreur en ne considérant que le cas d'une entrée est égale à 0.

Pour le code à répétition, cette entrée 0 est transmise $2s+1$ fois : l'entrée est le mot de code $C_0, \cdots, C_{2s}$ à $n = 2s+1$ caractères, tous égaux à 0. Pour chacun des 0 du mot d'entrée, la sortie y_j qui correspond vaut 1 avec probabilité p.

Notons $S_{2s+1} = \sum_{j=0}^{2s} y_j$ le nombre de 1 reçus lorsque le mot composé de $2s+1$ valeurs 0 est transmis. La probabilité d'erreur après prise de décision majoritaire (probabilité d'erreur par mot) est donnée par :

$$Pr\left(S_{2s+1} \geqslant s+1\right)$$

Remarque préliminaire. La probabilité d'erreur par mot s'écrit :

$$Pr\left(S_{2s+1} \geqslant s+1\right) = Pr\left(\frac{S_{2s+1}}{2s+1} \geqslant \frac{s+1}{2s+1}\right)$$

Lorsque $n \to \infty$, le terme $\frac{s+1}{2s+1}$ tend vers $1/2$ alors que, d'après la loi des grands nombres, la moyenne empirique $\frac{S_{2s+1}}{2s+1}$ tend vers $p < 1/2$, ainsi $\lim_{n\to\infty} Pr\left(S_{2s+1} \geqslant s+1\right) = 0$.

Calcul de la probabilité d'erreur après décodage. La probabilité pour que la somme $S_{2s+1} = \sum_{j=0}^{2s} y_j$ soit égale à k (entier compris entre 0 et $2s+1$) vaut :

$$Pr\left(S_{2s+1} = k\right) = C_{2s+1}^{k} p^k \left(1-p\right)^{2s+1-k}$$

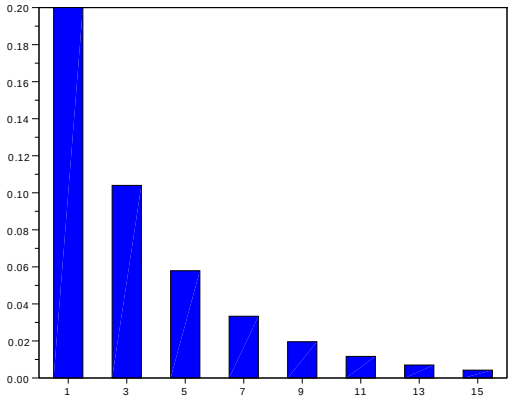


Figure 5.6: Performance d'un code à répétition sur un CBS pour $p = 1/5$ et une longueur des mots de code variant de 1 à 15.

La probabilité d'erreur est la probabilité de décider 1, cela se produit lorsque le nombre de 1 reçu est supérieur à $s + 1$:

$$Pr\left(S_{2s+1} \geq s+1\right) = \sum_{k=s+1}^{2s+1} C_{2s+1}^{k} p^k \left(1-p\right)^{2s+1-k}$$

La figure (5.6) représente cette probabilité d'erreur en fonction de la longueur des mots de code pour $p = 1/5$.

Ainsi, un code à répétition permet de réduire autant qu'on le souhaite la probabilité d'erreur. Le prix à payer est le suivant. Supposons que le canal puisse transmettre au maximum un symbole par seconde. Initialement, le débit d'information était de 1 bit/s. Après codage répétitif, il n'est plus que de $1/n = 1/(2s + 1)$ bit/s. Pour réduire à une valeur arbitrairement petite la probabilité d'erreur, il a donc fallu réduire dans les mêmes proportions le débit de transmission.

5.3.2 Théorème du codage canal

Contrairement à ce que pourrait laisser penser l'exemple du code à répétition, il n'est pas nécessaire de réduire à 0 le débit pour réduire à 0 la probabilité d'erreur.

Le résultat étonnant établit par Shannon est que, sous réserve que le débit reste inférieur à la capacité du canal, il est possible de réduire à une valeur arbitrairement petite la probabilité d'erreur sans réduire le débit de transmission. Le moyen permettant d'arriver à ce résultat s'appelle le codage canal.

Bien que, en général, les codes qui permettent d'arriver à ce genre de résultats soient beaucoup plus complexes qu'un code à répétition, tous utilisent d'une manière ou d'une autre une forme de répétition.

Notations

- X_n l'ensemble des suites $\mathbf{x}$ de longueur n en entrée du canal (écrites dans un alphabet d'entrée I).
- Y_n l'ensemble des suites $\mathbf{y}$ de longueur n en sortie du canal (écrites dans un alphabet de sortie J).
- $p(\mathbf{y}|\mathbf{x})$ la probabilité de transition du canal pour les séquences de longueur n (probabilité d'observer la séquence $\mathbf{y}$ lorsque la séquence $\mathbf{x}$ est en entrée).

Codeur canal. En entrée du canal, on utilise un sous-ensemble de $M = 2^k$ mots dans l'ensemble X_n des séquences de longueurs n ($k < n$). A chacun de ces M états, le codeur canal associe un mot de code composé d'une suite de n caractères (les caractères de l'alphabet d'entrée du canal, ici typiquement 0 ou 1).

Hypothèses

1. La source d'information est simple et de loi uniforme (copies indépendantes d'une v.a. uniformément distribuée sur l'alphabet utilisé).

   Si la source possède 2 états, son extension d'ordre k est une v.a. uniforme à $M = 2^k$ états et les k v.a. sont indépendantes et toutes de loi uniforme.

   Avant leur entrée dans le canal, ces k valeurs sont transformées, par une opération déterministe (codeur canal), en une suite plus longue de $n > k$ valeurs. Ainsi, chacun de ces n valeurs porte en moyenne $R = k/n$ bits d'information. R est le rendement du code.

2. Le canal est sans mémoire à entrées et sorties discrètes. Pour un canal sans mémoire $p(\mathbf{y}|\mathbf{x}) = \prod_{j=1}^{n} p(y_j|x_j)$.

Décodeur au sens du maximum du vraisemblance. A chacune des séquences $\mathbf{y}$ de longueur n en sortie du canal, le décodeur canal associe l'un des mots du code, c'est-à-dire l'un des états de la source étendue. La règle de décision qui minimise la probabilité d'erreur est celle du maximum de vraisemblance (cf. TD 8) :

On décide le mot $\mathbf{x_m}$ si

$$p(\mathbf{y}|\mathbf{x_m}) > p(\mathbf{y}|\mathbf{x_{m'}}) \, \forall m' \neq m$$

Le second théorème de Shannon ou théorème du codage canal donne une limite sur l'entropie de la source en dessous de laquelle le codage peut garantir un taux d'erreur aussi faible que nécessaire pour peu que la longueur des mots du code soit suffisamment grande.

La démonstration de ce résultat repose sur l'évaluation des performances de codes par bloc en utilisant la technique dite du "codage aléatoire".

Majoration de la probabilité d'erreur pour un mot Le mot m étant en entrée, la probabilité d'erreur $P_{e(m)}$ est la probabilité de décider $m' \neq m$, c'est-à-dire la probabilité qu'il existe un m' tel que $p(\mathbf{y}|\mathbf{x_{m'}}) > p(\mathbf{y}|\mathbf{x_m})$.

Notons

$$\phi_m(\mathbf{y}) = \begin{cases} 1 & \text{s'il existe un } m' \text{ tel que } p(\mathbf{y}|\mathbf{x_{m'}}) > p(\mathbf{y}|\mathbf{x_m}). \\ 0 & \text{sinon.} \end{cases}$$

La probabilité d'erreur lorsque le mot m est en entrée s'écrit :

$$P_{e(m)} = \sum_{\mathbf{y}\in Y_n} P\left(\mathbf{y}|\mathbf{x}_m\right)\phi_m\left(\mathbf{y}\right)$$

Majoration pour tout $s > 0$:

$$\phi_m\left(\mathbf{y}\right) \leq \left[\frac{\sum_{m'\neq m} P\left(\mathbf{y}|\mathbf{x}_{m'}\right)^{\frac{1}{1+s}}}{P\left(\mathbf{y}|\mathbf{x}_m\right)^{\frac{1}{1+s}}}\right]^s$$

Le second membre étant positif (probabilités positives) l'inégalité est immédiate pour $\phi_m\left(\mathbf{y}\right) = 0$.

Pour $\phi_m\left(\mathbf{y}\right) = 1$, il y a erreur de détection, c'est-à-dire qu'il existe un m' tel que $p\left(\mathbf{y}|\mathbf{x_{m'}}\right) > p\left(\mathbf{y}|\mathbf{x_m}\right)$, le numérateur est donc supérieur au dénominateur et le second membre est supérieur à 1. □

De cette majoration de $\phi_m\left(\mathbf{y}\right)$ découle celle de la probabilité d'erreur :

$$P_{e(m)} \leq \sum_{\mathbf{y}\in Y_n}\left\{P\left(\mathbf{y}|\mathbf{x}_m\right)^{\frac{1}{1+s}}\left[\sum_{m'\neq m} P\left(\mathbf{y}|\mathbf{x}_{m'}\right)^{\frac{1}{1+s}}\right]^s\right\}, s > 0$$

Introduction du codage aléatoire. L'idée consiste à considérer un ensemble probabilisé de codes pour lequel une majoration simple de la probabilité d'erreur moyenne peut être obtenue pour ensuite en déduire l'existence d'un code au sein de cet ensemble dont la probabilité d'erreur est également inférieure à cette borne.

Les mots du code sont issus de tirages indépendants selon une loi $P(\mathbf{x})$ (définie sur l'espace des séquences d'entrée).

Lorsque les mots de code sont aléatoires, les quantités $P\left(\mathbf{y}|\mathbf{x}_m\right)$ qui en dépendent sont des variables aléatoires indépendantes (pour différentes valeurs de m).

Majoration de la probabilité d'erreur pour un ensemble probabilisé de codes

$$\begin{aligned}
\bar{P_e} = \mathbb{E}P_{e(m)} &\leq \mathbb{E}\sum_{\mathbf{y}\in Y_n}\left\{P\left(\mathbf{y}|\mathbf{x}_m\right)^{\frac{1}{1+s}}\left[\sum_{m'\neq m} P\left(\mathbf{y}|\mathbf{x}_{m'}\right)^{\frac{1}{1+s}}\right]^s\right\}, 0 < s \\
\text{(Linéarité intégrale)} &= \sum_{\mathbf{y}\in Y_n}\mathbb{E}\left\{P\left(\mathbf{y}|\mathbf{x}_m\right)^{\frac{1}{1+s}}\left[\sum_{m'\neq m} P\left(\mathbf{y}|\mathbf{x}_{m'}\right)^{\frac{1}{1+s}}\right]^s\right\}, 0 < s \\
\text{(Mots indépendants)} &= \sum_{\mathbf{y}\in Y_n}\left\{\mathbb{E}\left[P\left(\mathbf{y}|\mathbf{x}_m\right)^{\frac{1}{1+s}}\right]\mathbb{E}\left[\left(\sum_{m'\neq m} P\left(\mathbf{y}|\mathbf{x}_{m'}\right)^{\frac{1}{1+s}}\right)^s\right]\right\}, 0 < s
\end{aligned}$$

Pour $0 < s < 1$ la fonction $f(x) = x^s$ est concave et l'inégalité de Jensen permet d'écrire la majoration :

$$\begin{aligned}
\bar{P_e} &\leq \sum_{\mathbf{y}\in Y_n}\left\{\mathbb{E}\left[P\left(\mathbf{y}|\mathbf{x}_m\right)^{\frac{1}{1+s}}\right]\left[\mathbb{E}\sum_{m'\neq m} P\left(\mathbf{y}|\mathbf{x}_{m'}\right)^{\frac{1}{1+s}}\right]^s\right\}, 0 < s < 1 \\
\text{(Linéarité intégrale)} &= \sum_{\mathbf{y}\in Y_n}\left\{\mathbb{E}\left[P\left(\mathbf{y}|\mathbf{x}_m\right)^{\frac{1}{1+s}}\right]\left[\sum_{m'\neq m}\mathbb{E}\left(P\left(\mathbf{y}|\mathbf{x}_{m'}\right)^{\frac{1}{1+s}}\right)\right]^s\right\}, 0 < s < 1
\end{aligned}$$

La quantité

$$\mathbb{E}\left(P\left(\mathbf{y}|\mathbf{x}_{m'}\right)^{\frac{1}{1+s}}\right) = \sum_{\mathbf{x}\in X_n} P(\mathbf{x})P\left(\mathbf{y}|\mathbf{x}\right)^{\frac{1}{1+s}}$$

est la même pour tous les mots m' d'où la majoration de la probabilité d'erreur moyenne pour un ensemble probabilisé de codes :

$$\bar{P_e} \leq (M-1)^s \sum_{\mathbf{y} \in Y_n} \left[\sum_{\mathbf{x} \in X_n} P(\mathbf{x}) P\left(\mathbf{y}|\mathbf{x}\right)^{\frac{1}{1+s}} \right]^{1+s}, 0 < s < 1$$

Cette majoration permet de déduire qu'il existe un code pour lequel cette borne est également un majorant de sa probabilité d'erreur.

Nous allons maintenant montrer que, pour peu que $R < C$, cette borne tend vers 0 lorsque la longueur des mots croît.

Borne pour un canal sans mémoire et une source simple. Pour trouver une forme simple de cette borne très générale, prenons maintenant en compte les hypothèses particulières qui sont faites ici :

- Le canal est sans mémoire :

$$p\left(\mathbf{y}|\mathbf{x}\right) = \prod_{j=1}^{n} p\left(y_j|x_j\right)$$

- La source est simple :

$$p\left(\mathbf{x}\right) = \prod_{j=1}^{n} p\left(x_j\right)$$

Sous ces hypothèses simplificatrices, la majoration devient :

$$\bar{P_e} \leq (M-1)^s \sum_{\mathbf{y} \in Y_n} \left[\sum_{\mathbf{x} \in X_n} \prod_{j=1}^{n} p\left(x_j\right) p\left(y_j|x_j\right)^{\frac{1}{1+s}} \right]^{1+s}, 0 < s < 1$$

Soit, en écrivant les sommes sur les alphabets d'entrée I et de sortie J du canal :

$$\bar{P_e} \leq (M-1)^s \left\{ \sum_{j \in J} \left[\sum_{k \in I} p_k p(j|k)^{\frac{1}{1+s}} \right]^{1+s} \right\}$$

En utilisant la majoration $M - 1 < M = 2^{nR}$ ($R = k/n$ le rendement du code), on a :

$$\bar{P_e} \leq 2^{-n[-sR + E_0(s, \{p_k\})]}$$

avec

$$E_0\left(s, \{p_k\}\right) = -\log_2 \left[\sum_{j \in J} \left[\sum_{k \in I} p_k p(j|k)^{\frac{1}{1+s}} \right]^{1+s} \right]$$

Ainsi, il existe un code dont la probabilité d'erreur par mot P_e est majorée comme suit :

$$P_e \leq 2^{-nE(R)} \text{ avec } E(R) = \max_{s, \{p_k\}} \left[-sR + E_0\left(s, \{p_k\}\right) \right]$$

Nous allons maintenant calculer cette borne dans le cas particulier du canal binaire symétrique et montrer qu'elle décroît pour s'annuler en $R = C$.

Calcul explicite de la borne pour un canal binaire symétrique. Pour simplifier la maximisation, nous supposons que la loi d'entrée est uniforme[2] de sorte que la maximisation ne porte que sur le paramètre s.

$$E_0\left(s,\{p_k\}\right) = -\log_2\left[\sum_{j=0}^{1}\left[\sum_{k=0}^{1} p_k p(j|k)^{\frac{1}{1+s}}\right]^{1+s}\right]$$

Pour $p_0 = p_1 = 1/2$:

$$E_0\left(s,\{p_k\}\right) = -\log_2\left[\left[\frac{1}{2}p(0|0)^{\frac{1}{1+s}} + \frac{1}{2}p(0|1)^{\frac{1}{1+s}}\right]^{1+s} + \left[\frac{1}{2}p(1|0)^{\frac{1}{1+s}} + \frac{1}{2}p(1|1)^{\frac{1}{1+s}}\right]^{1+s}\right]$$

Pour un CBS : $p(1|1) = p(0|0) = 1-p$ et $p(1|0) = p(0|1) = p$:

$$\begin{aligned} E_0\left(s,\{p_k\}\right) &= -\log_2\left[\frac{1}{2^s}\left[p^{\frac{1}{1+s}} + (1-p)^{\frac{1}{1+s}}\right]^{1+s}\right] \\ &= s - (1+s)\log_2\left[p^{\frac{1}{1+s}} + (1-p)^{\frac{1}{1+s}}\right] \end{aligned}$$

Pour éviter le calcul analytique fastidieux, il est possible de calculer numériquement le maximum

$$E(R) = \max_{0<s<1}\left\{s(1-R) - (1+s)\log_2\left[p^{\frac{1}{1+s}} + (1-p)^{\frac{1}{1+s}}\right]\right\}$$

et de tracer son évolution en fonction de R : la figure (5.7) représente l'évolution de la fonction $E(R)$ en fonction de R pour différentes valeurs de la capacité du canal[3].

[2] Cette hypothèse est naturelle puisque l'on sait que, pour un canal binaire symétrique, la capacité est atteinte pour une loi d'entrée uniforme. En maximisant seulement sur s, on trouvera ainsi une borne qui, même si elle n'était pas la plus fine, permet de conclure sur ce cas particulier.

[3] On a $-sR + E_0\left(s,\{p_k\}\right) = s(1-R) - sH_{\frac{1}{1+s}}$ où $H_\lambda = (1-\lambda)^{-1}\log_2\sum_k p_k^\lambda$ est l'entropie de Rényi ; H_λ est une fonction décroissante de λ avec $H_0 = \log_2 2 = 1$ et $H_\infty = -\log_2\max(p,1-p)$. En $R = C$, $-sR + E_0\left(s,\{p_k\}\right)|_{R=C} = s\left(H_1 - H_{\frac{1}{1+s}}\right)$, le minimum de $H_{\frac{1}{1+s}}$ est atteint en $s = 0$, point en lequel la fonction $-sR + E_0$ s'annule.

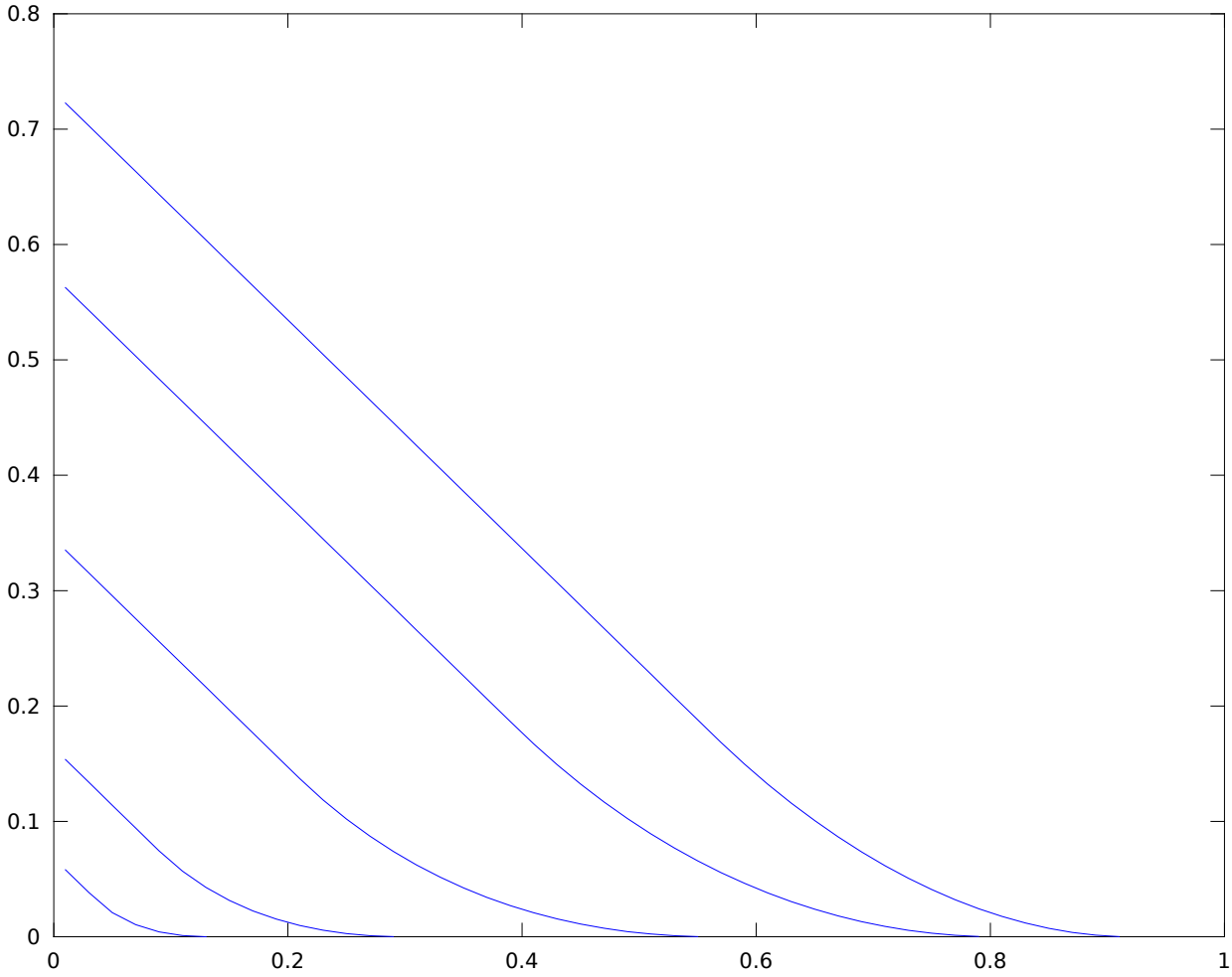


Figure 5.7: Borne $E(R)$ pour un canal binaire symétrique et les valeurs de capacités $C = 0.13$; 0.3 ; 0.56 ; 0.8 ; 0.92. La fonction $E(R)$ est positive, décroissante et s'annule en $R = C$ valeur du rendement au delà de laquelle la probabilité d'erreur par mot ne tend plus vers zéro lorsque la taille des mots de code croît.

Code octave pour tracer la figure (5.7).

```
% Merci à Rodrigo Cabral pour ce morceau de code
p=0.01:0.02:0.99;
s=0.01:0.02:0.99;
r=0.01:0.02:0.99;
invps=1./(1+s);
expon=zeros(length(p),length(r),length(s));
max_exp=zeros(length(p),length(r));
for i=1:length(p)
    for j=1:length(r)
        for k=1:length(s)
        expon(i,j,k)=s(k)*(1-r(j))
                    -(1+s(k))*log2((p(i)^invps(k))+((1-p(i))^(invps(k))));
        end
    max_exp(i,j)=max(expon(i,j,:));
    end
end
max_exp(max_exp<=0)=NaN;
capacity=1+(p.*log2(p)+(1-p).*log2(1-p));
f=max_exp';
hold;
plot(r,f(:,1)); capacity(1)
plot(r,f(:,2)); capacity(2)
plot(r,f(:,5)); capacity(5)
plot(r,f(:,10)); capacity(10)
plot(r,f(:,15)); capacity(15)
```

Commentaire sur le second théorème de Shannon. Le résultat remarquable établi par Shannon garantit que pour $R < C$ il existe un code dont la probabilité d'erreur est aussi faible que souhaité pour peu que la longueur des mots n soit suffisante.

Lorsque n croît, $k = nR$ croît de manière proportionnelle, cela signifie qu'il est possible de réduire la probabilité d'erreur à une valeur arbitrairement faible à rendement constant, ce qui est très différent de ce qui se produit pour un simple code à répétition pour lequel le rendement décroît lorsque la longueur des mots croît.

Chapter 6

Codage canal en pratique

Contents

Tout codage canal repose sur la répétition. Celle-ci permet de rendre la détection fiable et l'on sait, d'après le second théorème de Shannon, que cette fiabilité peut ne pas se faire au détriment du rendement, elle impose seulement que les mots du code soient de longueur suffisante.

Ce chapitre est consacré aux principes du codage par bloc.

F_2. Les codes abordés ici sont binaires : les mots du code sont écrits dans l'alphabet $F_2 = \{0, 1\}$. On munit F_2 de 2 opérations :

$\boxed{\oplus}$ le OU EXCLUSIF (addition modulo 2) :

$\oplus$	0	1
0	0	1
1	1	0

$\boxed{\& \text{ ou } \cdot}$ le ET (produit) :

&	0	1
0	0	0
1	0	1

$(F_2, \oplus)$ est une groupe abélien, $(F_2, \oplus, \cdot)$ est un corps commutatif.

6.1 Codage par bloc

Un code de rendement $R = k/n$ produit, à partir de k bits d'information, un mot de code composé de $n > k$ valeurs binaires.

Le codage est une application de F_2^k dans F_2^n qui à toute séquence binaire $U \in F_2^k$ de longueur k associe une séquence binaire $V \in F_2^n$ de longueur n.

La répartition des 2^k mots du code dans l'espace des séquences de longueur n, et plus particulièrement la distribution des distances entre mots (le spectre des distances) conditionnent les performances du code. La distance minimale entre les mots (distance minimale du code) est particulièrement importante.

Exemple. Pour le code à répétition $n = 3$ fois, le code compte $2^{k=1} = 2$ mots (000) et (111) parmi les $2^{n=3} = 8$ séquences possibles et le décodage au sens du maximum de vraisemblance s'effectue selon la règle :

- (000), (001), (010), (100): détection et correction vers (000)
- (111), (110), (101), (011): détection et correction vers (111)

La distance minimale $d_{\min}$ de ce code vaut $d_{\min} = 3$, c'est la distance de Hamming entre les mots (000) et (111).

Les séquences (000), (001), (010), (100) sont détectées de manière exacte en (000), leur distance de Hamming par rapport à (000) est inférieure ou égale à $1 = \lfloor (d_{\min} - 1)/2 \rfloor$.

Il en est de même pour les séquences (111), (110), (101), (011) par rapport au mot (111).

Ce code corrige au plus $\lfloor d_{\min} - 1)/2 \rfloor = 1$ erreur.

6.2 Codage linéaire par bloc

Un code en bloc est linéaire si les 2^k mots du code forment un sous espace vectoriel de F_2^n.

Ce sous espace vectoriel est de dimension k dans F_2^n de dimension n.

On note $C(n, k)$ un code bloc linéaire avec des mots de longueur n construits à partir de k bits informatifs.

6.2.1 Matrice génératrice.

Un code linéaire est caractérisé par sa matrice génératrice G.

Soit $\mathbf{m} \in F_2^k$ une séquence de k bits informatifs[1], on peut la décomposer dans une base $\{\mathbf{e}_i\}_{i=1..k}$ de F_2^k : $\mathbf{m} = \oplus_{i=1}^{k} m_i \mathbf{e}_i$. Pour un code linéaire, les mots sont une fonction g linéaire de $\mathbf{m}$: $g(\mathbf{m}) = \oplus_{i=1}^{k} m_i g(\mathbf{e}_i)$. Les k séquences $g(\mathbf{e}_i)$ peuvent être décomposées dans une base $\{\mathbf{e'}_j\}_{j=1..n}$ de F_2^n : $\mathbf{e}_i = \oplus_{j=1}^{n} g_{ij} \mathbf{e'}_j$. Les mots du code sont générés par :

$$\mathbf{c} = g(\mathbf{m}) = \mathbf{mG}$$

où

$$\mathbf{G} = \begin{bmatrix} g_{11} & \cdots & g_{1n} \\ \vdots & & \\ g_{k1} & \cdots & g_{kn} \end{bmatrix}$$

est appelée matrice génératrice du code.

Remarques :

- Les lignes de $\mathbf{G}$ sont des mots du code. Tout mot du code est combinaison linéaire des lignes.
- La matrice $\mathbf{G}$ n'est pas unique, elle dépend des bases choisies pour F_2^k et F_2^n.
- Il est toujours possible d'écrire la matrice génératrice sous une forme dite systématique :

  $$\mathbf{G} = \begin{bmatrix} \mathbf{I}_{k\times k} & \mathbf{P}_{k\times(n-k)} \end{bmatrix}$$

  de sorte que les mots du code s'écrivent :

  $$\mathbf{c} = \mathbf{mG} = \mathbf{m} \begin{bmatrix} \mathbf{I}_{k\times k} & \mathbf{P}_{k\times(n-k)} \end{bmatrix} = \begin{bmatrix} \mathbf{m} & \mathbf{mP} \end{bmatrix}$$

  Pour une telle forme, les k premières valeurs d'un mot sont les bits porteurs d'information tandis que les $n - k$ valeurs restantes constituent la redondance.

[1] La coutume en codage consiste à représenter les mots par des vecteurs ligne contrairement à l'usage en algèbre.

6.2.2 Matrice de contrôle de parité.

La matrice de contrôle de parité d'un code bloc linéaire est la matrice $(n-k) \times n$ définie par

$$\mathbf{G}\mathbf{H}^T = \mathbf{0}_{k\times(n-k)}$$

Si $\mathbf{G}$ est sous forme systématique :

$$\mathbf{H} = \left[\begin{array}{cc} \mathbf{P}^T_{k\times(n-k)} & \mathbf{I}_{(n-k)\times(n-k)} \end{array} \right]$$

En effet

$$\begin{aligned} \mathbf{G}\mathbf{H}^T &= \left[\begin{array}{cc} \mathbf{I}_{k\times k} & \mathbf{P}_{k\times(n-k)} \end{array} \right] \left[\begin{array}{cc} \mathbf{P}^T_{k\times(n-k)} & \mathbf{I}_{(n-k)\times(n-k)} \end{array} \right]^T \\ &= \mathbf{P}_{k\times(n-k)} \oplus \mathbf{P}_{k\times(n-k)} \\ &= \mathbf{0}_{k\times(n-k)} \end{aligned}$$

Les mots de code $\mathbf{c} = \mathbf{m}\mathbf{G}$ appartiennent au noyau de la matrice de parité : $(\mathbf{m}\mathbf{G})\,\mathbf{H}^T = \mathbf{0}$.

6.2.3 Syndrome.

Il est facile de détecter des erreurs en calculant le produit $\mathbf{s} = \mathbf{r}\mathbf{H}^T$ de la séquence reçue $\mathbf{r}$ par la matrice de parité, le résultat de cette opération est appelé le syndrome, c'est une séquence de longueur $n-k$.

L'entrée du canal est un mot du code $\mathbf{c}$, la sortie diffère de l'entrée en les positions des erreurs. Notons $\mathbf{p}$ la séquence composée d'un 1 aux positions des erreurs et de 0 ailleurs. La séquence $\mathbf{r}$ en sortie du canal est la somme modulo 2 du mot $\mathbf{c}$ et de la séquence des erreurs : $\mathbf{r} = \mathbf{c} \oplus \mathbf{p}$, le calcul du syndrome donne

$$\mathbf{r}\mathbf{H}^T = [\mathbf{c} \oplus \mathbf{p}]\,\mathbf{H}^T = \mathbf{c}\mathbf{H}^T \oplus \mathbf{p}\mathbf{H}^T = \mathbf{p}\mathbf{H}^T$$

Le syndrome est la somme des colonnes de la matrice de parité d'indices égaux aux positions des erreurs.

- Si le syndrome est nul, la séquence reçue est un mot du code (pas nécessairement le bon).
- Si le syndrome est non nul, il est certain qu'une erreur s'est produite.
- Des structures algébriques adaptées permettent de remonter du syndrome aux positions les plus probables d'un certain nombre d'erreurs et ainsi de les corriger avec un fort taux de succès. Une condition nécessaire pour cela est que le nombre de configurations possibles du syndrome (2^{n-k}) soit supérieur ou égal au nombre de configurations possibles du motif des erreurs $\mathbf{p}$ que l'on veut corriger. Par exemple, si l'on veut corriger une erreur sur un mot de longueur $n = 7$, il faut $n - k \geq 3$.

6.2.4 Distance minimale

La détection et la correction de certaines configurations d'erreurs sont possibles du fait que seule une fraction des séquences binaires de longueur n appartient à l'ensemble des mots du code.

La procédure de détection qui minimise la probabilité d'erreur par mot étant la recherche du mot de code à distance minimale de la séquence observée en sortie du canal, plus les mots sont espacés les uns des autres meilleure est la capacité de correction d'un code. Lorsque le canal est peu perturbé, la probabilité d'erreur est dominée par les confusions entre les mots les plus proches ; ainsi, la distance minimale joue un rôle essentiel et constitue une caractéristique très importante d'un code.

Cette distance minimale s'écrit :

$$d_{\min} = \min_{i\neq j} d_H \ \ (\mathbf{c}_i, \mathbf{c}_j) = \min_{i\neq j} \sum_{k=1}^{n} \mathbf{c}_{i_k} \oplus \mathbf{c}_{j_k}$$

où $\mathbf{c}_{i_k}$ désigne la composante k du mot i.

Ce calcul se simplifie en remarquant que, l'espace des mots du code étant un sous espace vectoriel de F_2^n, la somme de deux mots est également un mot. Ainsi, $\mathbf{c}_{i_k} \oplus \mathbf{c}_{j_k}$ est la composante k du mot $\mathbf{z} = \mathbf{c}_i \oplus \mathbf{c}_j$.

On définit le poids de Hamming $P_\mathbf{z}$ d'une séquence binaire $\mathbf{z} = [z_1, \cdots, z_n]$ comme le nombre de bits à 1 dans la séquence :

$$P_\mathbf{z} = \sum_{k=1}^{n} z_k$$

Ainsi, pour calculer la distance minimale, il suffit de calculer le poids de Hamming de l'ensemble des mots du code et de prendre le minimum.

6.3 Quelques exemples de codes élémentaires.

Contrôle de parité $C(3,2)$**.** Partant de $k = 2$ bits informatifs, ce code ajoute un bit de parité égal à leur somme modulo 2 pour produire des mots $\left[\begin{array}{ccc} c_0 & c_1 & c_2 \end{array}\right]$ de longueur $n = 3$ tels que $c_2 = c_0 \oplus c_1$:

$$\begin{array}{ccc} 00 & \longrightarrow & 000 \\ 01 & \longrightarrow & 011 \\ 10 & \longrightarrow & 101 \\ 11 & \longrightarrow & 110 \end{array}$$

En choisisant $\mathbf{e}_0 = [10]$ et $\mathbf{e}_1 = [01]$ pour base de F_2^k et $\mathbf{e'}_0 = [100]$, $\mathbf{e'}_1 = [010]$, $\mathbf{e'}_2 = [001]$ pour base de F_2^n, la matrice génératrice devient :

$$\mathbf{G} = \left[\begin{array}{ccc} 1 & 0 & 1 \\ 0 & 1 & 1 \end{array}\right] = \left[\begin{array}{cc} \underbrace{\left[\begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array}\right]}_{\mathbf{I}_{k\times k}=\mathbf{I}_{2\times 2}} & \overbrace{\left[\begin{array}{c} 1 \\ 1 \end{array}\right]}^{\mathbf{P}_{k\times(n-k)}=\mathbf{P}_{2\times 1}} \end{array}\right]$$

Pour ce code, la matrice de contrôle de parité est :

$$\mathbf{H} = \left[\begin{array}{ccc} 1 & 1 & 1 \end{array}\right] = \left[\begin{array}{cc} \underbrace{\left[\begin{array}{cc} 1 & 1 \end{array}\right]}_{\mathbf{P}^T_{2\times 1}} & \overbrace{\left[\begin{array}{c} 1 \end{array}\right]}^{\mathbf{I}_{1\times 1}} \end{array}\right]$$

et les mots du code vérifient $\mathbf{cH}^T = \mathbf{0}$. C'est-à-dire :

$$\left[\begin{array}{ccc} c_0 & c_1 & c_2 \end{array}\right] \left[\begin{array}{c} 1 \\ 1 \\ 1 \end{array}\right] = 0 \leftrightarrow c_0 \oplus c_1 \oplus c_2 = 0 \leftrightarrow c_2 = c_0 \oplus c_1$$

Performance de ce code de contrôle de parité. Si la séquence en sortie de canal est $\mathbf{r}$, on calcule le syndrome $\mathbf{s} = \mathbf{rH}^T$ et

- Si le syndrome $\mathbf{s}$ est non nul, il est certain qu'une erreur s'est produite.
- Si le syndrome $\mathbf{s}$ est nul, $\mathbf{r}$ est un mot du code, probablement celui qui a été placé en entrée du canal mais pas certainement. Par exemple, si le mot 011 est en entrée et que 2 erreurs se produisent, l'une en première position et l'autre en deuxième position, la sortie du canal est 101 qui est aussi un mot du code. Dans une configuration de ce type, le syndrome est nul et il n'existe aucun moyen de détecter la présence d'erreurs.

Code de Hamming (TD 9, 10). Introduit par Hamming en 1950, le code de Hamming $(7, 4)$ encode 4 bits de données en 7 valeurs binaires (ajout de 3 bits de parité). Ce code est capable de détecter et de corriger une erreur quelque soit sa position dans le mot de longueur 7. Sa matrice de parité est donnée par :

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

Chapter 7

Quelques mots sur l'approche algorithmique

7.1 Complexité de Lempel-Ziv

On considère

- un alphabet fini A, par exemple $A = \{0; 1\}$ ou $A = \{a; b; c; \cdots z\}$.
- Une séquence composée de n caractères $s = s(1), \cdots s(n)$ avec $s(i) \in A$.

On note

- $l(s)$ la longueur de la séquence s,
- Λ la séquence vide, de longueur $l(\Lambda) = 0$.
- $s(i, j)$ la sous séquence $s(i), \cdots s(j)$ avec la convention $s(i, j) = \Lambda$ pour $i > j$

Une séquence r est un préfixe de s s'il existe $j < l(s)$ tel que $r = s(1, j)$.

7.1.1 Reproductibilité

Une séquence $s = s(1, n)$ est reproductible à partir du préfixe $s(1, j)$ (avec $1 \leq j < n$) si elle peut être obtenue par copie récursive en commençant à une position p située dans le préfixe ($1 \leq p \leq j$), autrement dit si $s(j+1, n)$ est une sous séquence de $s(1, n-1)$. Pour que cela soit possible, il faut qu'il existe une position p dans le préfixe ($p < j$) telle que $s(j + 1, n) = s(p, p + l(j + 1, n) - 1)$.

On note $s(1, j) \rightarrow s$ le fait que s est reproductible à partir de $s(1, j)$.

Exemple : $0.0.1. \rightarrow 0.0.1.0.1.0.1.0.$ Considérons la séquence
$s = 0.0.1.0.1.0.1.0$ de longueur $l(s) = 8$ et de son préfixe de longueur 3 :
$r = 0.0.1.$

On cherche à générer les 5 caractères qui suivent le préfixe (l'extension $s(4, 8)$ en commençant une copie à une position p située dans le préfixe ($1 \leq p \leq 3$).

En partant de $p = 1$: le quatrième caractère $s(4) = 0$ peut être obtenu par copie du premier $s(1)$ puisque $s(4) = 0 = s(1)$. Poursuivre signifierait obtenir $s(5) = 1$ par copie de $s(2)$, cette opération est impossible car $s(5) \neq s(2)$. s ne peut pas être générée en partant de la position $p = 1$. Le même test doit être réitéré pour toutes les positions p comprises entre 1 et 3 (dans le préfixe).

En partant de $p = 2$, on a bien

$s(4) = 0 = s(2)$, puis

$s(5) = 0 = s(3)$, ici la copie a atteind la fin du préfixe (—$s(3)$—), néanmoins elle peut se poursuivre de manière récursive en copiant s(4) (qui était déjà une copie), ainsi :

$s(6) = 0 = s(4)$, puis

$s(7) = 0 = s(5)$, puis finalement

$s(8) = 0 = s(6)$.

Il est donc possible de générer l'extension $s(4, 8)$ en débutant une copie dans le préfixe en position $p = 2$: $s(1, 3) \to s(1, 8)$.

Exemple : $0.1. \nrightarrow 0.1.1.0.$ Considérons la séquence

$s = 0.1.1.0.$ de longueur $l(s) = 4$ et de son préfixe de longueur 2 :

$r = 0.1.$

On cherche à générer les 2 caractères qui suivent le préfixe (l'extension $s(3, 4)$ en commençant une copie à une position p située dans le préfixe ($1 \leq p \leq 2$).

En partant de $p = 1$: le troisième caractère $s(3) = 1$ diffère de $s(1) = 0$, aucune copie n'est possible.

En partant de $p = 2$: le troisième caractère $s(3) = 1 = s(2)$ peut être généré par copie, autrement dit $0.1. \to 0.1.1.$.

Cependant, la copie ne peut pas se poursuivre car $s(4) = 0 \neq s(3) = 1$.

Il n'existe donc aucune position p dans le préfixe qui permette de générer la séquence complète : $s = 0.1.1.0.$ n'est pas reproductible à partir du préfixe $r = 0.1$, on note $0.1. \nrightarrow 0.1.1.0.$.

7.1.2 Productibilité

Une séquence $s = s(1, n)$ est productible à partir du préfixe $s(1, j)$ (on note $s(1, j) \Rightarrow s(1, n)$ avec $j < n$) si $s = s(1, n - 1)$ est reproductible à partir du même préfixe.

Par définition, une séquence reproductible est donc productible mais toutes les séquences productibles ne sont pas reproductibles.

Exemple : $0.1. \Rightarrow 0.1.1.0.$ Considérons à nouveau la séquence

$s = 0.1.1.0.$ de longueur $l(s) = 4$ et de son préfixe de longueur 2 :

$r = 0.1.$

Nous avons vu que $0.1. \nrightarrow 0.1.1.0.$. (non reproductible) mais $0.1. \to 0.1.1.$ donc $0.1. \Rightarrow 0.1.1.0.$.

7.1.3 Histoire exhaustive

Pour toute séquence s, le premier caractère peut être produit à partir de la séquence vide :

$\Lambda \Rightarrow s(1, 1)$

Ce processus de production peut toujours se poursuivre :

$s(1, 1) \Rightarrow s(1, l_1)$ avec $l_1 = l_{\star 1} + 1$ où $l_{\star 1}$ est le plus grand l tel que $s(1, 1) \to s(1, l)$.

$s(1, l_j) \Rightarrow s(1, l_{j+1})$ avec $l_{j+1} = l_{\star j} + 1$ où $l_\star j$ est le plus grand l tel que $s(1, l_j) \to s(1, l_{j+1})$.

Comme on a toujours $s(1, j) \Rightarrow s(1, j + 1)$, le nombre d'étape pour produire toute séquence s est inférieur ou égal à $l(s)$.

A toute séquence, on peut associer une histoire (non unique) des productions qui la génère :

$\Lambda \Rightarrow s(1, 1) \Rightarrow s(1, l_1) \cdots s(1, l_{t-1}) \Rightarrow s(1, l_t)$

Chaque histoire correspond à une décomposition de la séquence s en t sous séquences :

$s = s(1, l_1).s(l_1 + 1, l_2) \cdots s(l_{t-1} + 1, l_t)$

A chaque étape, la copie qui génère le plus long prolongement, est dite exhaustive. En choisisant à chaque étape la version exhaustive, l'histoire obtenu est appelée histoire exhaustive, c'est la décomposition la plus courte (avec le moins de composantes).

Exemple.

7.1.4 Complexité

Le nombre de composante de l'histoire exhaustive est une mesure possible de la complexité de la séquence s.

Complexité moyenne d'une séquence i.i.d.

7.2 Codage de Lempel-Ziv

Les techniques de copier-coller évoquées ci-dessus pour établir une mesure de complexité conduisent directement à des procédures de codage.

Pour quelques détails voir la page "Complexité de Lempel-Ziv" sur fr.wikipedia.org.

Chapter 8

Echantillonnage

Contents

Quelques remarques préliminaires

Retrouver une fonction à partir de ses valeurs en certains points est parfois possible moyennant certaines hypothèses, c'est par exemple le cas pour les polynômes. Les résultats sur l'échantillonnage remontent à Edmund Taylor Whittaker (1915), Harry Nyquist (1928), Vladimir Kotelnikov (1933) et Claude Shannon (1949)[1].

Les signaux en provenance du monde physique dépendent d'une ou de plusieurs variables continues, typiquement le temps ou/et l'espace. Leur traitement sur un calculateur passe par une étape de numérisation. Celle-ci consiste à transformer une fonction de une ou plusieurs variables réelles en une suite dénombrable de valeurs réelles (par exemple en prélevant les valeurs prises par le signal à différents instants), c'est l'échantillonnage ; puis, à quantifier les valeurs réelles ainsi produites pour les rendre représentables sur une machine, c'est la quantification.

[1] Voici quelques références historiques :

E. T Whittaker — 1915. On the functions which are represented by the expansions of the interpolation-theory. Edinburgh University

Harry Nyquist — 1928. Certain topics in telegraph transmission theory. Transactions of the American Institute of Electrical Engineers (Reproduction dans Proceedings of the IEEE Volume 90, Issue 2, Feb 2002, p. 280-305).

V.A. Kotelnikov — 1933. On the transmission capacity of the 'ether' and of cables in electrical communications. Proceedings of the first All-Union Conference on the technological reconstruction of the communications sector and the development of low-current engineering. Moscow, 1933.

J. M. Whittaker — 1935. Interpolatory Function Theory. Cambridge University Press. New York, Macmillan, 1935.

D. Gabor — 1946. Theory of communication. J. IEE (London) ,1946,November ,Vol. 93 ,Part III ,No. 26 ,Page 429-457

Claude E. Shannon — 1949. Communication in the presence of noise. Proc. Institute of Radio Engineers vol. 37 (1): 10–21. http://www.stanford.edu/class/ee104/shannonpaper.pdf

Claude E. Shannon — 1949. C. E. Shannon (1949, reprinted 1998). The Mathematical Theory of Communication. Urbana, IL:University of Illinois Press.

8.1 Echantillonnage

Les systèmes physiques possèdent de l'inertie, ils ne peuvent répondre à des entrées arbitrairement rapides : leur réponse diminue lorsque la fréquence augmente, on dit qu'ils sont passe-bas et donc, au moins approximativement, à bande limitée. Ainsi, un cas particulièrement utile est celui des signaux à bande limitée pour lesquels une reconstruction exacte est possible.

On considère $x(t)$, un signal continu, stable (sommable) et passe-bas, c'est-à-dire qui ne contient aucune énergie en dehors de la bande $\left[-\frac{B}{2},+\frac{B}{2}\right]$ (sa transformée de Fourier $\widehat{x}(\nu)$ est nulle en dehors de cet intervalle).

8.1.1 Théorème d'échantillonnage

Pour un tel signal, la transformée de Fourier est continue et à support borné, ainsi $\widehat{x}(\nu)$ est sommable et la synthèse de $x(t)$ s'écrit :

$$x(t)=\int_{-\frac{B}{2}}^{+\frac{B}{2}}\widehat{x}(\nu)e^{i2\pi\nu t}d\nu$$

La fonction $p(\nu)=\sum_{k\in\mathbb{Z}}\widehat{x}\left(\nu-kB\right)$ obtenue en périodisant avec la période B la fonction $\widehat{x}(\nu)$ a pour coefficients de Fourier :

$$\widehat{p}_n=\frac{1}{B}\int_{-\frac{B}{2}}^{+\frac{B}{2}}p(\nu)e^{-i2\pi\nu\frac{n}{B}}d\nu=\frac{1}{B}\int_{-\frac{B}{2}}^{+\frac{B}{2}}\widehat{x}(\nu)e^{-i2\pi\nu\frac{n}{B}}d\nu=\frac{1}{B}x\left(-\frac{n}{B}\right)$$

Si la suite des échantillons $\left\{x\left(\frac{n}{B}\right)\right\}_{n\in\mathbb{Z}}$ est sommable, on a :

$$p(\nu)=\frac{1}{B}\sum_{k\in\mathbb{Z}}x\left(-\frac{k}{B}\right)e^{+i2\pi\nu\frac{k}{B}}$$

soit

$$x(t)=\int_{-\frac{B}{2}}^{+\frac{B}{2}}\widehat{x}(\nu)e^{i2\pi\nu t}d\nu=\int_{-\frac{B}{2}}^{+\frac{B}{2}}p(\nu)e^{i2\pi\nu t}d\nu=\frac{1}{B}\sum_{k\in\mathbb{Z}}x\left(-\frac{k}{B}\right)\int_{-\frac{B}{2}}^{+\frac{B}{2}}e^{i2\pi\nu\frac{k}{B}}e^{i2\pi\nu t}d\nu$$

$$\int_{-\frac{B}{2}}^{+\frac{B}{2}}e^{i2\pi\nu\frac{k}{B}}e^{i2\pi\nu t}d\nu=B\mathrm{sinc}\left[\pi B\left(t+\frac{k}{B}\right)\right]$$

d'où

$$\begin{aligned}x(t)&=\sum_{k\in\mathbb{Z}}x\left(-\frac{k}{B}\right)\mathrm{sinc}\left[\pi B\left(t+\frac{k}{B}\right)\right]\\&=\sum_{k\in\mathbb{Z}}x\left(\frac{k}{B}\right)\mathrm{sinc}\left[\pi B\left(t-\frac{k}{B}\right)\right]\end{aligned}$$

Si un signal ne contient aucune fréquence en dehors de l'intervalle $\left[-\frac{B}{2},+\frac{B}{2}\right]$, une reconstruction parfaite du signal analogique est possible à partir des échantillons $\left\{x\left(kT\right)\right\}_{k\in\mathbb{Z}}$ sous la condition $T<\frac{1}{B}$, l'expression du signal analogique en fonction de ses échantillons est :

$$x(t)=\sum_{k\in\mathbb{Z}}x\left(kT\right)\mathrm{sinc}\left[\frac{\pi}{T}\left(t-kT\right)\right]\tag{8.1}$$

La figure 8.1 illustre la reconstruction selon la formule de synthèse 8.1. Il est clair que la reconstruction est imparfaite car les hypothèses qui permettent la reconstruction ne sont pas vérifiées : le signal est à durée limitée, sa transformée de Fourier ne peut donc pas être à support borné.

En pratique, les signaux sont tous de durée limitée, le résultat de Shannon est exact sous des hypothèses qui ne sont jamais vérifiées. De ce point de vue, le problème de l'échantillonnage et de reconstruction est un problème d'approximation.

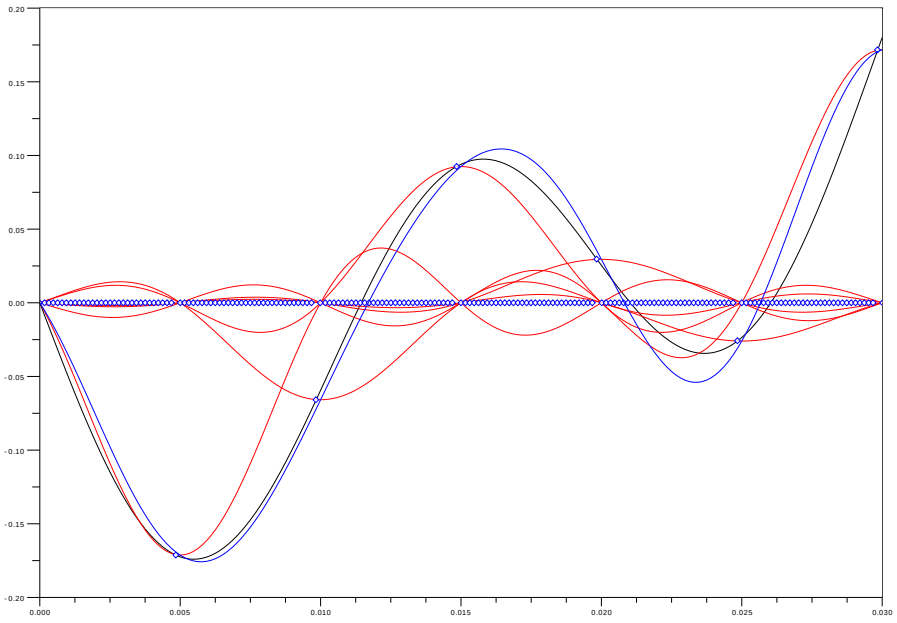


Figure 8.1: Un signal composé de trois fréquences pures (en noir), les sinus cardinaux pour chaque échantillon (en rouge) et leur somme selon la formule de reconstruction (en bleu)

8.1.2 Repliement de spectre

Que se produit-il si l'on échantillonne toutes les T secondes un signal $x(t)$ dont la bande n'est pas limitée à $\left[-\frac{1}{2T}, +\frac{1}{2T}\right]$?

On suppose toujours que $\sum_{k\in\mathbb{Z}} |x(kT)| < +\infty$, le signal reconstruit s'écrit :

$$T\sum_{k\in\mathbb{Z}} x\left(kT\right)\int_{-\frac{1}{2T}}^{+\frac{1}{2T}} e^{i2\pi\nu(t-kT)}d\nu = T\int_{-\frac{1}{2T}}^{+\frac{1}{2T}} e^{i2\pi\nu t}\left[\sum_{k\in\mathbb{Z}} x\left(kT\right)e^{-i2\pi\nu kT}\right]d\nu$$

En supposant $\widehat{x}(\lambda)$ intégrable et $x(t)$ continu, on a en Fourier :

$$\left[T\sum_{k\in\mathbb{Z}} x\left(kT\right)e^{-i2\pi\nu kT}\right]1_{\left[-\frac{1}{2T},+\frac{1}{2T}\right]}(\nu)$$

$$=$$

$$\begin{aligned}
&\left[\sum_{k\in\mathbb{Z}} T\left(\int_{\mathbb{R}}\widehat{x}(\lambda)e^{i2\pi\lambda kT}d\lambda\right)e^{-i2\pi\nu kT}\right]1_{\left[-\frac{1}{2T},+\frac{1}{2T}\right]}(\nu)\\
=&\left[\sum_{k\in\mathbb{Z}} T\left(\sum_{n\in\mathbb{Z}}\int_{-\frac{1}{2T}}^{+\frac{1}{2T}}\widehat{x}(\lambda+\frac{n}{T})e^{i2\pi\left(\lambda+\frac{n}{T}\right)kT}d\lambda\right)e^{-i2\pi\nu kT}\right]1_{\left[-\frac{1}{2T},+\frac{1}{2T}\right]}(\nu)\\
=&\left[\sum_{k\in\mathbb{Z}} T\left(\int_{-\frac{1}{2T}}^{+\frac{1}{2T}}\left[\sum_{n\in\mathbb{Z}}\widehat{x}(\lambda+\frac{n}{T})\right]e^{i2\pi\lambda kT}d\lambda\right)e^{-i2\pi\nu kT}\right]1_{\left[-\frac{1}{2T},+\frac{1}{2T}\right]}(\nu)
\end{aligned}$$

$\left[\sum_{n\in\mathbb{Z}}\widehat{x}(\lambda+\frac{n}{T})\right]$ est continu, périodique de période $1/T$ et intégrable sur l'intervalle $\left[-\frac{1}{2T},+\frac{1}{2T}\right]$ d'où :

$$\sum_{n\in\mathbb{Z}}\widehat{x}(\lambda+\frac{n}{T}) = \sum_{k\in\mathbb{Z}} T\left(\int_{-\frac{1}{2T}}^{+\frac{1}{2T}}\left[\sum_{n\in\mathbb{Z}}\widehat{x}(\lambda+\frac{n}{T})\right]e^{i2\pi\lambda kT}d\lambda\right)e^{-i2\pi\nu kT}$$

La transformée de Fourier du signal recontruit est donc égale à :

$$1_{\left[-\frac{1}{2T},+\frac{1}{2T}\right]}(\nu)\sum_{n\in\mathbb{Z}}\widehat{x}(\lambda+\frac{n}{T})$$

En pratique, on s'assure que le signal que l'on échantillonne ne comporte aucune fréquence en dehors de la bande $\left[-\frac{1}{2T},+\frac{1}{2T}\right]$ en le filtrant avant échantillonnage par un filtre passe-bas. Ce filtre qui évite le repliement de spectre est appelé filtre antirepliement.

8.1.3 Echantillonneur réalisable

Prélever un échantillon. Que signifie prélever la valeur en t_0 d'un signal $x(t)$? Comment formaliser mathématiquement cette opération ? Est-elle physiquement réalisable ? Si non, comment peut-elle être approchée pratiquement ?

Physiquement, la valeur $x\left(0\right)$ de $x(t)$ au point $t=0$ ($t=0$ est choisi pour simplifier les notations) peut être approchée en intégrant la fonction $x(t)$ sur un voisinage de 0, par exemple de la manière suivante :

$$x\left(0\right) \approx \frac{1}{\epsilon}\int_{-\epsilon}^{0} x(\alpha)d\alpha$$

En effet, pour $\epsilon << 1$, si la fonction $x(t)$ est suffisamment régulière, $x(t) \approx x\left(0\right)$ sur l'intervalle d'intégration. En notant $\delta^{(\epsilon)} = \frac{1}{\epsilon}1_{[0,\epsilon]}$, on a :

$$x\left(0\right) \approx \int_{\mathbb{R}}\delta^{(\epsilon)}(\alpha)x(-\alpha)d\alpha$$

Plus la durée d'intégration est courte, plus cette formule est précise mais, malheureusement, plus l'énergie présente dans la fenêtre d'intégration est faible ($\int_{-\frac{\epsilon}{2}}^{+\frac{\epsilon}{2}} |x(t)|^2 \to 0$ pour $\epsilon \to 0$).

Physiquement, il n'est pas possible de prendre une durée d'intégration arbitrairement courte car la mesure requière de l'énergie. Pas d'énergie, pas de mesure. La précision des appareils de mesure étant limitée et le bruit omniprésent, l'énergie doit être présente en quantité suffisante, autrement dit, la durée d'intégration doit être suffisamment grande.

Prélever une suite régulière d'échantillons. A partir du signal $x(t)$, l'échantillonneur réalisable produit la suite d'échantillons

$$z_k = z\left(kT\right) = \epsilon^{-1} \int_{kT-\epsilon}^{kT} x(\alpha) d\alpha \text{ avec } \epsilon << T.$$

Les valeurs ainsi obtenues traduisent le comportement moyen du signal sur la durée ϵ au voisinage de kT : les fluctuations sur cette durée sont gommées par cette intégration.

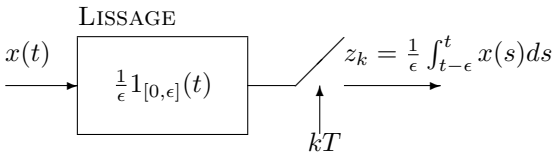


Figure 8.2: L'échantillonneur réalisable est équivalent à un échantillonnage idéal du signal lissé par la porte .

Représentation équivalente de l'échantillonneur réalisable. Une autre manière de voir ce point est la suivante : la suite des valeurs z_k peut également être obtenue en échantillonnant de manière idéale une version lissée du signal $x(t)$ (figure 8.2).

Soit $z(t)$ la version de $x(t)$ filtrée par $\delta^{(\epsilon)}(t) = \epsilon^{-1} 1_{[0,\epsilon]}$, on a :

$$z(t) = \delta^{(\epsilon)}(t) \star x(t) = \int_{\mathbb{R}} \frac{1}{\epsilon} 1_{[0,\epsilon]}(\alpha) x(t-\alpha) d\alpha = \frac{1}{\epsilon} \int_0^{\epsilon} x(t-\alpha) d\alpha$$

c'est-à-dire

$$z(t) = \delta^{(\epsilon)}(t) \star x(t) = \frac{1}{\epsilon} \int_{t-\epsilon}^{t} x(\alpha) d\alpha$$

L'échantillonnage aux points kT en sortie de ce filtre passe-bas donne bien les échantillons z_k :

$$z\left(kT\right) = \frac{1}{\epsilon} \int_{t-\epsilon}^{t} x(\alpha) d\alpha \Big|_{t=kT} = \frac{1}{\epsilon} \int_{kT-\epsilon}^{kT} x(\alpha) d\alpha$$

Impact de l'échantillonnage non idéal. Dans le domaine fréquentiel, la relation de convolution $z(t) = \delta^{(\epsilon)}(t) \star x(t)$ donne :

$$\left|\widehat{z(\nu)}\right| = \left|\widehat{\delta^{(\epsilon)}}(\nu)\right| \left|\widehat{x}(\nu)\right|$$

avec

$$\left|\widehat{\delta^{(\epsilon)}}(\nu)\right| = \left|e^{-i\pi\nu\epsilon} \text{sinc}\left(\pi\nu\epsilon\right)\right| = \left|\text{sinc}\left(\pi\nu\epsilon\right)\right|$$

La convolution par $\epsilon^{-1} 1_{[0,\epsilon]}$ pondère les fréquences du signal $x(t)$ par un sinus cardinal, les basses fréquences passent mieux que les hautes fréquences : le signal a subi un filtrage dit passe-bas, $z(t)$ est une version lissée de $x(t)$.

En pratique, pour une application donnée, il faut que la durée d'intégration soit suffisamment petite par rapport à la période d'échantillonnage et au taux de distorsion acceptable.

8.1.4 Nombre de degré de liberté d'un signal.

Bien qu'il ne soit pas strictement possible de considérer un signal limité à la fois en temps et en fréquence, cette hypothèse est approximativement vérifiée en pratique : un morceau de musique est de durée limitée et l'on considère également que sa bande est limitée à l'intervalle $[-20 \text{ khz}, 20 \text{ khz}]$. Ceci signifie que la part de son énergie hors de cette bande est négligeable.

Vision intuitive de la dimension d'un signal. Une fréquence pure tronquée à la durée T se modélise par le produit d'une exponentielle complexe et d'une porte (fonction indicatrice de l'intervalle $\left[-\frac{T}{2}, +\frac{T}{2}\right]$) :

$$e^{i2\pi\nu_0 t} 1_{\left[-\frac{T}{2}, +\frac{T}{2}\right]}(t)$$

ce n'est plus une fréquence pure, sa transformée de Fourier ne possède plus un support réduit à la seule fréquence ν_0 mais vaut :

$$T\text{sinc}\left[\pi\left(\nu - \nu_0\right)T\right]$$

La premier zéro de ce sinus cardinal est à distance $1/T$ de la fréquence ν_0. Deux sinus cardinaux espacés de $1/T$ sont orthogonaux. Si l'écart entre deux fréquences ν_0 et ν_0' est inférieur à $1/T$, elles ne sont pas séparables, $1/T$ est en quelque sorte la résolution fréquentielle pour des signaux de durée égale à T. Le nombre de fréquences séparables sur la bande B est de l'ordre de $\frac{B}{1/T} = BT$.

BT est le nombre de dégrés de liberté d'un signal de bande B et de durée T au sens où un tel signal peut être écrit (à une erreur faible près) comme combinaison linéaire de BT fonctions orthogonales.

Ce résultat (qui peut se démontrer de manière rigoureuse) rejoint le théorème d'échantillonnage : pour un signal de bande B, il faut, pour être à même de le représenter par une suite discrète de réels, prélever BT échantillons sur la durée T, c'est-à-dire une période d'échantillonnage $T/(BT) = 1/B$.

Part II

Annexes

Appendix A

Chiffres élémentaires

Les probabilités et les statistiques sont des outils adaptés à beaucoup d'opérations de maniement de l'information. Donnons quelques indices par des exemples en cryptographie et en cryptanalyse.

La cryptographie a pour objectif de réécrire un message sous une forme non déchiffrable pour une personne qui ignore le secret de son chiffrage (une clef).

La cryptanalyse vise à casser le chiffrement.

Pour des codes par substitution, l'analyse des fréquences d'apparition des différents caractères qui composent le message est une méthode de cryptanalyse. Par exemple, en français, le E représente environ 17% des lettres d'un texte, suivi du A (7%), du S etc. Une estimation fiable de ces fréquences dépend de la taille et de la représentativité de l'échantillon utilisé pour cette estimation. Dans un texte court, des déviations non négligeables peuvent apparaître. Un exemple d'estimation sur un texte de petite taille est donné par la figure (A)

A.0.1 Le chiffre de César.

Le chiffre de César procède par décalage circulaire dans l'alphabet $\{A, \cdots, Z\}$. Par exemple, pour un décalage de 2 : $A \to C, B \to D \cdots Y \to A, Z \to B$.

Ce chiffre doit son nom à Jules César qui l'utilisait entre autre (avec un décalage de 3) pour écrire à Ciceron.

Ce chiffrement par substitution (chaque lettre est remplacée par une autre lettre) à distance fixe dans l'alphabet est très faible du fait du peu de clefs possibles (26). Pour le décoder, il suffit de tester les 26 décalages.

Chiffrement. Le chiffrement est une substitution monoalphabétique.

La constante utilisée pour le décalage est aussi la clef.

La méthode de déchiffrement est la même (avec décalage opposé).

Cryptanalyse La méthode brute est suffisante : essayer chacune des 26 clefs.

Le plus souvent, une analyse des fréquences des lettres du message déchiffré avec les différentes clefs suivie d'un calcul de distance avec la fréquence des lettres de la langue du message suffit à l'identification de la clef.

Quelques cas particuliers célèbres :

Décalage de 1. C'est le chiffre d'Auguste.

Décalage de 2. C'est le code Hélène (LN) : pour $L \to N$, le décalage vaut 2.

Décalage de 13. ROT13 est le nom donné au chiffre de César pour un décalage de 13. Pour l'alphabet Latin, ROT13 est son propre inverse (en décalant à nouveau de 13, on retrouve l'original, $13 + 13 = 26 = 0[26]$).

ROT13 est utilisé sur des forums pour que des solutions d'énigmes ne sautent pas aux yeux. C'est un peu l'équivalent Usenet de l'écriture tête en bas des solutions dans les revues papiers.

```
Homme libre, toujours tu chériras la mer!
La mer est ton miroir; tu contemples ton âme
Dans le déroulement infini de sa lame,
Et ton esprit n'est pas un gouffre moins amer.

Tu te plais à plonger au sein de ton image;
Tu l'embrasses des yeux et des bras, et ton coeur
Se distrait quelquefois de sa propre rumeur
Au bruit de cette plainte indomptable et sauvage.

Vous êtes tous les deux ténébreux et discrets:
Homme, nul n'a sondé le fond de tes abîmes;
Ô mer, nul ne connaît tes richesses intimes,
Tant vous êtes jaloux de garder vos secrets!

Et cependant voilà des siècles innombrables
Que vous vous combattez sans pitié ni remords,
Tellement vous aimez le carnage et la mort,
Ô lutteurs éternels, ô frères implacables!
```

```
E. 17.2 %
S. 9.8 %
T. 9.1 %
A. 6.8 %
R. 6.7 %
O. 6.7 %
N. 6.5 %
U. 5.8 %
I. 5.6 %
L. 5.1 %
M. 4.7 %
D. 3.5 %
C. 2.3 %
P. 2.1 %
B. 1.9 %
V. 1.4 %
F. 1.1 %
G. 1.1 %
H. 0.7 %
X. 0.7 %
Q. 0.5 %
Z. 0.4 %
J. 0.4 %
Y. 0.2 %

Deux lettres manquent :
K. 0 %
W. 0 %
```

Figure A.1: Estimation de fréquences sur un texte court

A.0.2 Le chiffre de Vigenère.

Attribué à Blaise de Vigenère (diplomate français, 1523-1596), ce système de substitution polyalphabétique remplace chaque lettre par une autre (variable) obtenue à partir d'un ensemble de décalages qui constitue la clef.

Chiffrement. La clef est une suite de lettres, en utilisant l'équivalence lettre-chiffre

A	B	C	· · ·	Z
0	1	2	· · ·	25

, c'est aussi une suite de décalages. On chiffre la première lettre du message en lui ajoutant la première lettre de la clef (modulo 26), la deuxième lettre du message en lui ajoutant la deuxième lettre de la clef, etc. Si la longueur de la clef est inférieure à celle du message, on reprend la clef à son début après avoir utilisé son dernier caractère.

Exemple avec le message `QUEL BEAU CHIFFRE` **et la clef** `IMAG` $(8, 12, 0, 6)$

```
Message original : QUEL BEAU CHIFFRE
Clef périodisée  : IMAG IMAG IMAGIMA
Message chiffré  : YGER JQAA KTILNDE
```

En pratique, le chiffrement s'effectue avec une table dont les colonnes correspondent aux caractères à coder et les lignes aux caractères de la clef.

Déchiffrement. Il suffit de soustraire la clef au message en utilisant la même méthode que pour le chiffrement.

Cryptanalyse. Casser le chiffre de Vigenère est d'autant plus difficile que la clef est longue : lorsque la longueur de la clef est égale à celle du texte, cela est impossible, c'est le chiffre de Vernam, le seul système de cryptographie parfaitement sûr. A l'opposé, pour une clef de longueur 1, Le chiffre de Vigenère se réduit à celui de César dont la cryptanalyse est évidente. Entre ces deux extrêmes, une piste réside dans le fait que les lettres espacées de la longueur de la clef subissent le même décalage. Ainsi, si la longueur de la clef était connue, il suffirait de réaliser une analyse des fréquences (cf. cryptanalyse du chiffre de César) pour chacun des sous-messages obtenus en sous échantillonnant le message initial à la longueur de la clef. La question préalable est donc de déterminer la longueur de la clef, pour cela plusieurs pistes peuvent être explorées, par exemple :

Indice de coincidence. En français, la probabilité pour que deux lettres tirées au hasard dans un texte coïncident vaut environ 0.074 (indice de coïncidence 0,065 pour l'anglais.). En testant différentes longueurs de clef pour trouver celles pour lesquelles l'indice de coïncidence se rapproche le plus de cette valeur, on peut estimer la longueur de la clef.

Méthode de Kasiski. La méthode recherche les répétitions (typiquement d'au moins 3 lettres) dans le message chiffré, celles-ci peuvent provenir de deux occurrences d'une même séquence originale chiffrées avec la même portion de clef. L'écart entre les deux séquences est alors un multiple de la longueur de la clef.

Si l'on repère deux séquences différentes qui se répètent à distances d_1 et d_2, les diviseurs communs de d_1 et d_2 sont des longueurs possibles pour la clef.

Appendix B

Quantification et représentation des nombres.

Même échantillonné, un signal analogique n'est pas représentable sur un calculateur, il faut aussi discrétiser son amplitude pour que la valeur de chaque échantillon puisse être codée sur un nombre fini de bits, c'est la quantification.

Contrairement à l'échantillonnage, qui peut être réalisé sans perte sous l'hypothèse physiquement réaliste de bande limitée, la quantification engendre presque systématiquement une perte d'information, ce n'est pas une opération réversible.

Ce paragraphe aborde très brièvement les méthodes de quantification les plus courantes ainsi que la modélisation de l'erreur de quantification.

Il existe quelques manières habituelles de représenter les nombres signés en virgule fixe[1] : représentation de la grandeur et du signe ; représentation en complément à deux.

En virgule fixe , à la valeur signée x on associe la représentation binaire finie $a_n, a_{n-1}, \cdots, a_0, a_{-1}, \cdots, a_{-b}$ définit par :

Bit de signe	Représentation du Module	
	$a_{n-1}, \cdots, a_0$	$a_{-1}, \cdots, a_{-b}$
a_n	Partie entière	Partie fractionnaire
$a_n = 0$: positif $a_n = 1$: négatif	$\sum_{j=0}^{n-1} a_j 2^j$	$\sum_{j=-b}^{-1} a_j 2^j$

Le pas de quantification vaut $q = 2^{-b}$. La position de la virgule est conventionnelle (deux cas particuliers courants : les nombres entiers ($b = 0$) et les décimaux normalisés à 1 ($n = 0$)).

Grandeur et signe (GS). La représentation du module contient le codage binaire $\sum_{j=-b}^{n-1} a_j 2^j$ du module et a_n code le signe d'où $x = (-1)^{a_n} \sum_{j=-b}^{n-1} a_j 2^j$. Les valeurs représentables sont symétriquement réparties autour de 0 :

$$-\left(2^n - 2^{-b}\right) < -2^n + 2^{-b+1} < \cdots < +2^n - 2^{-b}$$

[1] Nous ne considérons que les représentations en virgule fixe qui, contrairement à ce que l'on pourrait penser sont les plus utilisées en pratique du fait de leur moindre complexité (donc moindre consommation des processeurs et moindre surface de silicium) et du fait que, en dessous de 16 bits, la représentation en virgule fixe offre une dynamique supérieure à la représentation flottante. Au-dessus de 24 bits, la dynamique de la représentation flottante devient très supérieure à celle de la représentation fixe.

Complément à 2 (C2). Identique au codage GS pour les nombres positifs (le bit de signe vaut 0 tandis que la ZRN (zone de représentation du nombre) code le module). Pour les négatifs, le bit de signe vaut toujours 1 tandis que la zone de représentation du nombre x représente le complément à 2^n du module : $\sum_{j=-b}^{n-1} a_j 2^j = 2^n - |x|$

$x \geqslant 0$	$0, a_{n-1}, \cdots, a_0, a_{-1}, \cdots, a_{-b}$ avec $x = \sum_{j=-b}^{n-1} a_j 2^j$	$x = \sum_{j=-b}^{n-1} a_j 2^j$
$x < 0$	$1, a_{n-1}, \cdots, a_0, a_{-1}, \cdots, a_{-b}$ avec $\|x\| = 2^n - \sum_{j=-b}^{n-1} a_j 2^j$	$x = -\|x\| = -2^n + \sum_{j=-b}^{n-1} a_j 2^j$
Pour tout x	$a_n, a_{n-1}, \cdots, a_0, a_{-1}, \cdots, a_{-b}$	$x = -a_n 2^n + \sum_{j=-b}^{n-1} a_j 2^j$

Remarque : le zéro est représenté une seule fois, le nombre de valeurs binaires étant pair, la représentation est asymétrique autour de 0. Les valeurs représentables sont :

$$-2^n < -2^n + 2^{-b} < \cdots < +2^n - 2^{-b}$$

Pour la représentation décimale ($n = 0$), $-1 < -2^n + 2^{-b} < \cdots < 1 - 2^{-b}$.

Exemple sur 3 bits			
	GS	*C2*	*Valeur*
+	011	011	$3/3$
+	010	010	$2/4$
+	001	001	$1/4$
+	000	000	$+0$
-	100		-0
-	101	111	$-1/4$ZRN représente $2^{n=0} - \left\|-\frac{1}{4}\right\| = 2^0 - 2^{-2} = 2^{-2} + 2^{-1} \rightarrow$ 11
-	110	110	$-2/4$
-	111	101	$-3/4$
-		100	-1

Opérations en virgule fixe pour le complément à deux. Le complément à deux est le plus utilisé pour ses bonnes propriétés en terme de calcul en particulier pour sa tolérance aux débordements (dépassement de la gamme des valeurs représentables). Examinons de plus près quelques propriétés de cette arithmétique circulaire.

Changement de signe. La complémentation d'une valeur binaire $b \in \{0,1\}$ est l'application $\{0,1\} \rightarrow \{0,1\}$ telle que $b \mapsto \overline{b} = 1 - b$. L'opposé de $x = -a_n 2^n + \sum_{j=-b}^{n-1} a_j 2^j$ s'écrit :

$$-x = a_n 2^n - \sum_{j=-b}^{n-1} a_j 2^j = -(1 - a_n) 2^n + 2^n - \sum_{j=-b}^{n-1} a_j 2^j$$
$$2^n - \sum_{j=-b}^{n-1} a_j 2^j = \left(2^n - 2^{-b} - \sum_{j=-b}^{n-1} a_j 2^j\right) + 2^{-b} = \sum_{j=-b}^{n-1} (1 - a_j) 2^j + 2^{-b}$$

C'est-à-dire :

$$-x = -(1 - a_n) 2^n + \sum_{j=-b}^{n-1} (1 - a_j) 2^j + 2^{-b}$$

Autrement dit, la représentation en complément à deux de l'opposé d'un nombre s'obtient en complémentant les bits puis en ajoutant 1 au mot binaire ainsi obtenu.

Addition et soustraction. Il n'y a rien de spécial à faire, tous les bits (y compris le bit de signe) sont traités de la même façon :

$$x+y=-x_n 2^n+\sum_{j=-b}^{n-1} x_j 2^j-y_n 2^n+\sum_{j=-b}^{n-1} y_j 2^j=-\left(x_n+y_n\right)2^n+\sum_{j=-b}^{n-1}\left(x_j+y_j\right)2^j$$

Pour la somme de plus de deux termes, les débordements temporaires sont autorisés tant que le résultat final ne déborde pas. Interprétation : les opérations se font modulo 2^{n+1}. Pour soustraire deux nombres, il suffit de prendre l'opposé et de sommer.

Débordements. —

- Lors de l'addition de deux nombres positifs, le débordement est indiqué par un changement de signe (le bit de signe du résultat vaut 1) et un bit de poids fort à 0.
- Lors de l'addition de deux nombres négatifs, le débordement est indiqué par un changement de signe (le bit de signe du résultat vaut 0) et un bit de poids fort à 1.
- Lors de l'addition de deux nombres de signes opposés, aucun débordement ne peut survenir.

Multiplication. Le produit de deux nombres codés sur b+1 bits donne un résultat sur 2b+1 bits et nécessite une troncature (re-quantification). Pour le calcul d'une somme de produit, les DSP (Digital Signal Processor) disposent de registres internes plus larges (typiquement le double) pour limiter l'impact du cumul des erreurs d'arrondi.

B.0.1 Types de quantification, erreur de quantification.

Lors du traitement de signaux analogiques, la quantification consiste à appliquer au signal une fonction ne prenant qu'un nombre fini de valeurs (fonction en escalier) pour passer d'une valeur analogique à l'une des valeurs discrètes représentables sur une machine. La quantification intervient également lors du traitement de signaux à amplitude quantifiée : puisque les opérations arithmétiques imposent de contraindre le format des résultats, une re-quantification est nécessaire.

Trois possibilités sont souvent implantées sur les DSP :

Arrondi. On associe à la valeur à quantifier la valeur discrète la plus proche (cf. figure B.1), cette méthode est la meilleure mais aussi la plus coûteuse. Nous notons l'erreur d'arrondi $e_A=Q_A\left(x\right)-x$

Troncature du module. Parfois utilisée pour éviter des problèmes de cycle limite. L'erreur engendrée par troncature du module $e_M=Q_M\left(x\right)-x$ (cf. figure B.1) varie entre $-q$ et $+q$. Elle se compose d'une partie systématique $-q/2\mathrm{sign}\left(x\right)$ ajoutée à une erreur de même nature que l'erreur d'arrondi.

Troncature du complément à deux. La moins coûteuse en calculs. Elle se compose d'une partie systématique $-q/2$ ajoutée à une erreur de même nature que l'erreur d'arrondi.

Bruit de quantification. L'erreur de quantification, aussi appelée bruit de quantification, c'est-à-dire l'écart entre la valeur (amplitude du signal) avant quantification et la valeur après quantification (amplitude quantifiée), est le plus souvent inconnu et l'on choisit une caractérisation statistique de cette erreur.

- Loi du bruit de quantification. Dans le cas de la quantification par arrondi, le seul que nous traitons ici, les valeurs de l'erreur sont comprises dans l'intervalle $[-q/2,+q/2]$. La modélisation classique consiste à supposer que cette erreur est une variable aléatoire à valeurs dans $[-q/2,+q/2]$. En l'absence d'information complémentaire, on choisit la loi "de plus grand désordre", celle qui maximise l'entropie : la loi uniforme sur

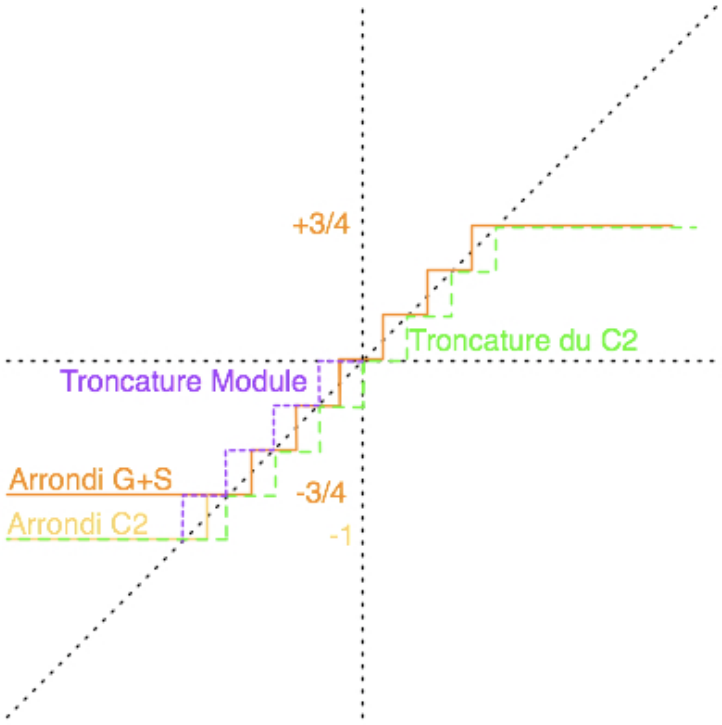


Figure B.1: Quantifications classiques illustrées dans le cas d'un codage sur 3 bits..

l'intervalle $[-q/2, +q/2]$.[2] En pratique, cette hypothèse est approximativement vérifiée lorsque l'amplitude du signal est beaucoup plus grande que le pas de quantification.

Sous cette hypothèse, le bruit de quantification est centré et sa variance vaut $q^2/12$.

[2]Exercice : montrer que la loi à densité de maximum d'entropie sur un support borné est la loi uniforme.

Bibliography

[1] R.B. ASH. Information Theory. Dover, 1990.

[2] H. ATLAN. L'organisation biologique et la théorie de l'information. La librairie du XXIème siècle - Seuil, février 2006.

[3] L. BRILLOUIN. La science et la théorie de l'information. Paris, Masson, 1959. (Reprint, 1988, 314 p., Broché, ISBN 2-87647-036-5.) Calcul des probabilités, 1966, suivi de Introduction à la théorie de l'information, 1966 , Reprint, 1992, 24,5 x 18 oblong, 320 pages, 2 titres en 1 volume, ISBN 2-87647-082-9.

[4] T.M. COVER, J.A. THOMAS. Elements of information theory. Second edition, 2006. Wiley.

[5] G. DUBERTRET. Initiation à la cryptographie. Vuibert 2002 (3ème édition)

[6] D. K. FADEEV. Zum Begriff der Entropie einer endlichen Wahrseheinlichkeiitsschemas, Arbeiten zur Informationstheorie I, Berlin, Deutscher Verlag der Wissensehaften, 1957, pp. 85-90.

[7] R.P. FEYNMAN. Leçons sur l'informatique. Odile Jacob, 1996.

[8] O. FRANCOIS. Notes de cours de Probabilités. Polycopié 1ère année. Ensimag.

[9] S. KULLBACK. Information Theory and Statistics. Dover, 1997

[10] D. J.C. MacKay. Information Theory, Inference, and Learning Algorithms. Copyright Cambridge University Press 2003. On-screen viewing permitted. Printing not permitted. http://www.cambridge.org/0521642981. See http://www.inference.phy.cam.ac.uk/mackay/itila/ for links.

[11] A. RÉNYI. Measures of information and entropy, in Proceedings of the 4th Berkeley Symposium on Mathematics, Statistics and Probability 1960, pp. 547–561.

[12] E. ROUBINE. Introduction à la théorie de la communication. Tome 3 : Théorie de l'information. MASSON 1970.