COURS TI EI2

Theorie de l'information

Codage source-canal

ESCPI-CNAM

Version 2.4

31 janvier 2007

Didier Le Ruyet

Introduction

Ce cours est consacré aux bases de communications numériques. Pour les lecteurs qui souhaitent approfondir leurs connaissances, nous recommandons le livre de Proakis [24] qui est le livre de référence dans le domaine. Nous ne traiterons que quelques points fondamentaux de la théorie de l'information, du codage de source et du codage de canal. Pour un traitement plus profond du sujet, nous recommandons la lecture des ouvrages de Battail en français [1], de Gallager [14], MacKay [20] et de Cover et Thomas [8].

Chapitre 1

Introduction

L'objectif fondamental d'un système de communication est de reproduire en un point de la chaîne de communication, soit exactement soit approximativement, un message sélectionné en un autre point (Claude Shannon 1948) [27].

Le problème qui se pose est que le canal est généralement bruité : comment obtenir une communication parfaite dans ces conditions ? Dans ce document, nous allons tenter de répondre à cette question.

Voici quelques exemples de canaux de transmission :
- une ligne téléphonique entre deux modems
- un canal radiomobile entre une station de base et un mobile
- un disque dur

Le dernier exemple montre que le mot *point* dans la phrase d'introduction peut signifier lieu ou temps.

La source génère un message à transmettre au destinataire. Celui-ci peut être analogique (parole, son, image, ...) ou numérique (données). Dans un système de communication numérique, le message analogique devra être converti en numérique avant traitement.

Exemple : le système de communication très simple suivant consiste à transmettre une image à un destinataire à travers un canal binaire symétrique.

Le canal binaire symétrique est décrit par le modèle de la figure 7.1. Il est défini par sa probabilité de transition $p = P(Y = 0|X = 1) = P(Y = 1|X = 0)$.

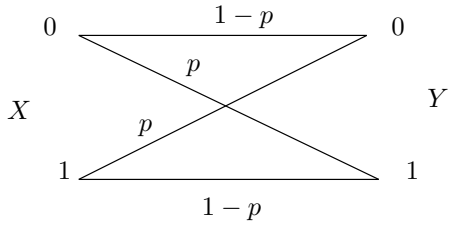


FIG. 1.1 – Canal binaire symétrique

Sur la figure 1.2, nous présentons un exemple d'image émise puis reçue par le destinataire en sortie d'un canal binaire symétrique pour $p = 0.1$. Dans un système de communication il faut protéger les bits d'information contre les erreurs de transmission tout en limitant le nombre de bits transmis dans le canal de transmission.

Sur la figure 1.3, nous présentons le synoptique d'un système de communication tel qu'il est décrit par Shannon.

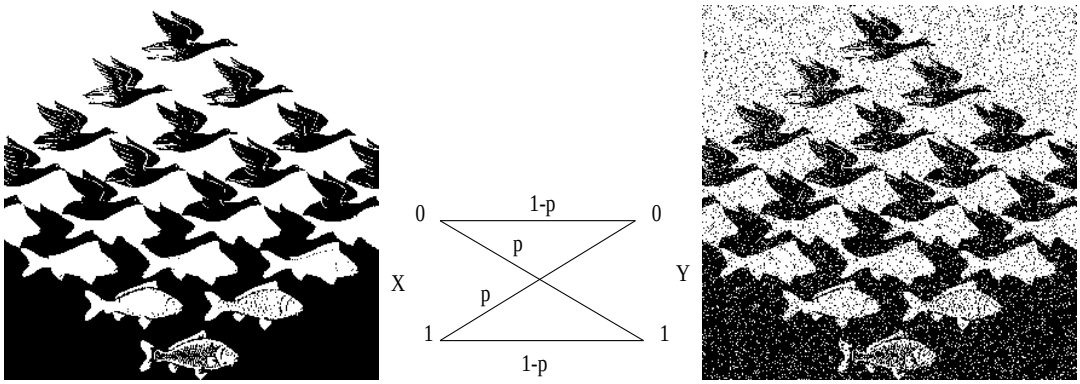


FIG. 1.2 – Image en entrée et en sortie du canal binaire symétrique.

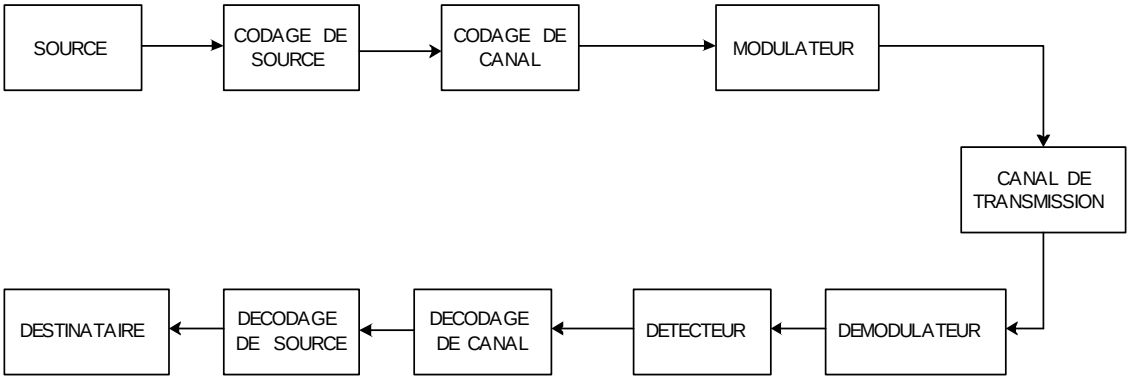


FIG. 1.3 – Synoptique d'une chaîne de communication.

L'objectif du codeur de source est de représenter le message avec le moins de bits possibles. Pour ce faire, il cherche à éliminer toute la redondance contenue dans le message de la source.

Le rôle du codage de canal est de protéger le message des perturbations du canal de transmission en ajoutant de la redondance au message compressé.

La modulation consiste à effectuer un codage dans l'espace euclidien, espace généralement adapté aux canaux rencontrés en pratique. Pour une modulation M-aire, on associe à chaque mot de g bits un signal $x_i(t), i = 1, \ldots, M$ de durée T choisi parmi les $M = 2^g$ signaux.

Le rôle du démodulateur est d'extraire les échantillons tout en maximisant le rapport signal à bruit. L'objectif du détecteur est de décider en faveur des symboles les plus probablement émis. Si le décodage de canal est à entrées pondérées, il faudra remplacer ce détecteur par un détecteur délivrant des informations pondérées (en général des logarithmes de rapport de vraisemblance).

La qualité d'un système de transmission est évaluée en calculant ou en mesurant la probabilité d'erreurs par bit d'information (ou par bloc d'information). Les deux autres paramètres importants d'un système de communication sont sa complexité et son occupation spectrale.

Chapitre 2

Introduction à la théorie de l'information

Where is the life we have lost in living ? Where is the wisdom we have lost in knowledge ? Where is the knowledge we have lost in information ? *T.S. Eliot, "The Rock"*

2.1 Rappels de probabilités

Soit une variable aléatoire X ayant pour espace de réalisations $A_X = \{x_1, x_2, \ldots, x_n\}$ (on parle aussi d'alphabet) avec les probabilités respectives $P_X = \{p_1, p_2, \ldots, p_n\}$ avec :

$$P(X = x_i) = p_i \quad p_i \geq 0 \quad \text{et} \quad \sum_{x_i \in A_X} p_i = 1 \tag{2.1}$$

Probabilité conjointe

Soit deux variables aléatoires X et Y ayant pour espace de réalisations respectif $A_X = \{x_1, x_2, \ldots, x_n\}$ et $A_Y = \{y_1, y_2, \ldots, y_m\}$

On appelle $P(X = x_i, Y = y_j)$ la probabilité conjointe des évènements $X = x_i$ et $Y = y_j$. On a bien entendu :

$$\sum_{x_i \in A_x} \sum_{y_j \in A_y} P(X = x_i, Y = y_j) = 1 \tag{2.2}$$

Probabilité marginale

Il est possible d'obtenir la probabilité $P(X = x_i)$ à partir de la probabilité conjointe $P(X = x_i, Y = y_j)$:

$$P(X = x_i) = \sum_{y_j \in A_y} P(X = x_i, Y = y_j) \tag{2.3}$$

Probabilité conditionnelle

$$P(X = x_i | Y = y_j) = \frac{P(X = x_i, Y = y_j)}{P(Y = y_j)} \tag{2.4}$$

De la même manière on a :

$$P(Y = y_j|X = x_i) = \frac{P(X = x_i, Y = y_j)}{P(X = x_i)} \tag{2.5}$$

Ainsi on a la relation

$$P(Y = y_j, X = x_i) = P(X = x_i|Y = y_j)P(Y = y_j) = P(Y = y_j|X = x_i)P(X = x_i) \tag{2.6}$$

Loi de Bayes

$$\begin{aligned} P(X = x_i|Y = y_j) &= \frac{P(Y = y_j|X = x_i)P(X = x_i)}{P(Y = y_j)} \\ &= \frac{P(Y = y_j|X = x_i)P(X = x_i)}{\sum_{x_k \in A_x} P(X = x_k, Y = y_j)} \\ &= \frac{P(Y = y_j|X = x_i)P(X = x_i)}{\sum_{x_k \in A_x} P(Y = y_j|X = x_k)P(X = x_k)} \end{aligned}$$

$P(X = x_i|Y = y_j)$ est appelée la probabilité a posteriori sur la réalisation de l'événement $X = x_i$ sachant la réalisation de l'événement $Y = y_j$. $P(Y = y_i)$ est appelée la probabilité a priori.

Indépendance

L'indépendance de deux variables aléatoires X et Y implique

$$P(X, Y) = P(X)P(Y) \tag{2.7}$$

et

$$P(X|Y) = P(X) \tag{2.8}$$

2.2 Remarques sur la notion d'information

La notion quantitative d'information associée à un message échangé entre un émetteur et un destinataire dans le language usuel est liée à :

- la véracité du message
- la connaissance *a priori* du message par le destinataire
- la compréhension du destinataire (problème de langue, ...)
- l'intérêt qu'y apporte le destinataire
- ...

Toutes ces considérations relèvent de la sémantique.

Dans la théorie de l'information, nous ne retiendrons qu'un aspect partiel du concept général d'information : la mesure quantitative d'information est une mesure de l'incertitude associée à un événement. Cette notion d'information est fondamentale dans l'étude des systèmes de communication.

2.3 Une mesure logarithmique de l'information

Une mesure de l'information associée à l'événement $X = x_i$ notée $h(x_i)$ doit satisfaire les propriétés suivantes [27] :

- $h(x_i)$ doit être continue pour $p(X = x_i)$ compris entre 0 et 1
- $h(x_i) = \infty$ si $P(X = x_i) = 0$
- $h(x_i) = 0$ si $P(X = x_i) = 1$: un événement certain n'apporte pas d'information

- $h(x_i) > h(y_j)$ si $P(Y = y_j) > P(X = x_i)$
- $h(x_i) + h(y_j) = h(x_i, y_j)$. La réalisation de 2 évènements indépendants $Y = y_j$ et $X = x_i$ apporte une quantité d'information égale à la somme des informations de ces 2 évènements $h(x_i)$ et $h(y_j)$.

La seule expression de la quantité d'information $h(x_i)$ associée à la réalisation de l'évènement $X = x_i$ satisfaisant les propriétés énumérées ci-dessus est la suivante :

$$h(x_i) = \log_2 \frac{1}{P(X = x_i)} = -\log_2 P(X = x_i) = -\log_2 p_i \tag{2.9}$$

On peut noter qu'avec cette définition, un évènement très probable transportera moins d'information qu'un évènement peu probable. Lorsque la base 2 est utilisée [1], l'unité de $h(x_i)$ est le Shannon (Sh). Lorsque le logarithme népérien est utilisé, l'unité est le Nat (*natural unit* en anglais).

Exemple 1 : soit une source discrète produisant des bits (0 ou 1) avec la probabilité $\frac{1}{2}$. La quantité d'information associée à la réalisation de l'événement $X = 0$ ou $X = 1$ est égale à :

$$h(0) = h(1) = -\log_2 \frac{1}{2} = 1 \text{ Sh} \tag{2.10}$$

Si cette source génère une séquence de n bits indépendants, il y a 2^n séquences différentes. Chacune de ces séquences se produit avec la probabilité $\frac{1}{2^n}$. L'information apportée par la réalisation d'une séquence particulière est égale à :

$$h(\text{séquence de } n \text{ bits}) = -\log_2 \frac{1}{2^n} = n \text{ Sh} \tag{2.11}$$

Considérons maintenant la réalisation de 2 évènements $X = x_i$ et $Y = x_j$. La quantité d'information associée est :

$$h(x_i, y_j) = \log_2 \frac{1}{P(X = x_i, Y = y_j)} = -\log_2 P(X = x_i, Y = y_j) \tag{2.12}$$

où $P(X = x_i, Y = y_j)$ est la probabilité conjointe des deux évènements.

La quantité d'information associée à la réalisation de l'événément $X = x_i$ conditionnellement à l'évènement $Y = y_j$ est la suivante :

$$h(x_i|y_j) = \log_2 \frac{1}{P(X = x_i|Y = y_j)} = -\log_2 P(X = x_i|Y = y_j) \tag{2.13}$$

A partir de la relation (2.6), on en déduit :

$$h(x_i, y_j) = h(x_i|y_j) + h(y_j) = h(y_j|x_i) + h(x_i) \tag{2.14}$$

Exemple 2 : on tire une carte au hasard dans un jeu de 32 cartes (4 couleurs : cœur, pique, carreau et trèfle - 8 valeurs : 7,8, 9, 10 valet dame roi as). Soit x l'événement "la carte tirée est un as de trèfle" et y l'événement "la carte tirée est un trèfle". Calculons $h(x)$, $h(y)$ et $h(x|y)$.

Comme

$$P(X = x) = \frac{1}{32} \qquad \text{et} \qquad P(Y = y) = \frac{1}{4} \tag{2.15}$$

Nous obtenons :

$$h(x) = -\log_2 \frac{1}{32} = 5 \text{ Sh} \qquad \text{et} \qquad h(y) = -\log_2 \frac{1}{4} = 2 \text{ Sh} \tag{2.16}$$

$$P(X = x|Y = y) = \frac{P(X = x, Y = y)}{P(Y = y)} = \frac{1/32}{1/4} = \frac{1}{8} \tag{2.17}$$

$$h(x|y) = -\log_2 P(X = x|Y = y) = -\log_2 \frac{1}{8} = 3 \text{ Sh} \tag{2.18}$$

[1] $\log_2 x = \frac{\ln x}{\ln 2}$

2.3.1 Information mutuelle

On définit l'information mutuelle comme la quantité d'information que la réalisation de l'événement $Y = y_j$ apporte sur l'événement $X = x_i$. Plus exactement, c'est la différence entre la quantité d'information associé à la réalisation de l'événement $X = x_i$ et la quantité d'information associé à la réalisation de l'événement $X = x_i$ conditionnellement à l'événement $Y = y_j$. Cette quantité d'information s'obtient comme suit :

$$\begin{aligned} i(x_i, y_j) &= h(x_i) - h(x_i|y_j) \\ &= \log_2 \frac{P(X = x_i|Y = y_j)}{P(X = x_i)} \end{aligned} \tag{2.19}$$

Si les deux événements sont indépendants, alors $P(X = x_i|Y = y_j) = P(X = x_i)$ et donc $i(x_i, y_j) = 0$. A l'opposé, si l'événement $X = x_i$ est équivalent à l'événement $Y = y_j$, alors $P(X = x_i|Y = y_j) = 1$ et $i(x_i, y_j) = h(x_i)$.

Comme on a la relation

$$\frac{P(X = x_i|Y = y_j)}{P(X = x_i)} = \frac{P(X = x_i, Y = y_j)}{P(X = x_i)P(Y = y_j)} = \frac{P(Y = y_j|X = x_i)}{P(Y = y_j)}$$

L'information que la réalisation de l'évènement $Y = y_j$ apporte sur l'évènement $X = x_i$ est identique à l'information que la réalisation de l'évènement $X = x_i$ apporte sur l'évènement $Y = y_j$.

$$i(x_i, y_j) = i(y_j, x_i) \tag{2.20}$$

On a également les relations suivantes :

$$\begin{aligned} i(x_i, y_j) &= h(x_i) - h(x_i|y_j) \\ &= h(x_i) + h(y_j) - h(x_i, y_j) \\ &= h(y_j) - h(y_j|x_i) \end{aligned}$$

Contrairement à $h(x_i)$, l'information mutuelle $i(x_i, y_j)$ peut être négative.

Suite de l'Exemple 2 : Calculons $i(x, y)$

$$\begin{aligned} i(x, y) &= h(x) - h(x|y) \\ &= 5 \text{ Sh} - 3 \text{ Sh} = 2 \text{ Sh} \end{aligned} \tag{2.21}$$

La quantité d'information que la réalisation de l'événement "la carte tirée est un trèfle" apporte sur l'événement "la carte tirée est un as de trèfle" est égale à 2 Sh.

L'information mutuelle est importante pour les communications en particulier lorsque l'on identifie X à l'entrée d'un canal de transmission et Y au signal correspondant à la sortie du canal de transmission.

2.4 Entropie et information mutuelle moyenne

Après s'être intéressé aux évènements individuels, nous allons maintenant déterminer l'entropie d'une source décrite par la variable aléatoire X ayant pour espace de réalisation $A_X = \{x_1, x_2, \ldots, x_n\}$ avec les probabilités respectives $P_X = \{p_1, p_2, \ldots, p_n\}$. n est la taille de l'alphabet. La quantité d'information moyenne ou entropie de la source est la moyenne des informations relatives à chaque réalisation de l'évènement $X = x_i$:

$$\begin{aligned} H(X) &= \sum_{i=1}^{n} p_i h(x_i) \\ &= \sum_{i=1}^{n} p_i \log_2 \frac{1}{p_i} \\ &= -\sum_{i=1}^{n} p_i \log_2 p_i \quad \text{en Sh/symbole} \end{aligned} \tag{2.22}$$

$H(X)$ mesure l'incertitude sur X.

Propriétés :

$$H(X) \geq 0 \quad \text{avec égalité si } p_i = 1 \text{ pour une valeur de } i \tag{2.23}$$

$$H(X) \leq \log_2 n \tag{2.24}$$

$$H(X) = H_{MAX}(X) = \log_2 n \qquad \text{si} \quad p_i = \frac{1}{n} \quad \forall i \tag{2.25}$$

L'entropie est donc maximale lorsque toutes les probabilités p_i sont égales.

Exemple : soit une source à 2 états x_0 et x_1 avec $p_0 = p$ et $p_1 = 1 - p$
L'entropie de cette source est la suivante :

$$H(X) \equiv H(p, 1-p) = -p \log_2 p - (1-p) \log_2 (1-p) \tag{2.26}$$

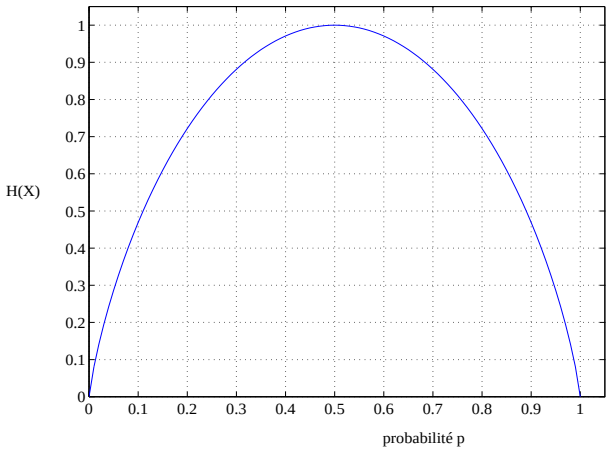


FIG. 2.1 – Entropie d'une source binaire

Ainsi, l'entropie de la source binaire est maximale (1 Sh/bit) lorsque $p = 0.5$

Considérons deux variables aléatoires X et Y ayant respectivement pour espace de réalisations $A_X = \{x_1, x_2, \ldots, x_n\}$ et $A_Y = \{y_1, y_2, \ldots, y_m\}$. L'entropie conjointe $H(X, Y)$ est définie comme

suit :

$$\begin{aligned} H(X,Y) &= \sum_{i=1}^{n}\sum_{j=1}^{m} P(X=x_i, Y=y_j) h(x_i, y_j) \\ &= -\sum_{i=1}^{n}\sum_{j=1}^{m} P(X=x_i, Y=y_j) \log_2 P(X=x_i, Y=y_j) \end{aligned} \tag{2.27}$$

Si les variables aléatoires X et Y sont indépendantes, l'entropie conjointe est égale à la somme des entropies $H(X)$ et $H(Y)$.

On peut aussi déterminer l'entropie conditionnelle $H(X/Y)$ qui détermine l'information sur X sachant l'observation Y à partir de $h(x_i|y_j)$:

$$\begin{aligned} H(X|Y) &= \sum_{i=1}^{n}\sum_{j=1}^{m} P(X=x_i, Y=y_j) h(x_i|y_j) \\ &= -\sum_{i=1}^{n}\sum_{j=1}^{m} P(X=x_i, Y=y_j) \log_2 P(X=x_i|Y=y_j) \end{aligned} \tag{2.28}$$

Les relations (2.6) ou (2.14) permettent d'exprimer l'entropie conjointe en fonction de l'entropie et l'entropie conditionnelle :

$$H(X,Y) = H(X) + H(Y|X) = H(Y) + H(X|Y) \tag{2.29}$$

L'incertitude sur X et Y est égale à l'incertitude sur X plus l'incertitude sur Y sachant X.

L'information mutuelle associée à la réalisation d'un événement peut également être étendue aux variables aléatoires X et Y.

$$\begin{aligned} I(X,Y) &= \sum_{i=1}^{n}\sum_{j=1}^{m} P(X=x_i, Y=y_j) i(x_i, y_j) \\ &= \sum_{i=1}^{n}\sum_{j=1}^{m} P(X=x_i, Y=y_j) \log_2 \frac{P(X=x_i|Y=y_j)}{P(X=x_i)} \\ &= \sum_{i=1}^{n}\sum_{j=1}^{m} P(X=x_i, Y=y_j) \log_2 \frac{P(X=x_i, Y=y_j)}{P(X=x_i)P(Y=y_j)} \end{aligned} \tag{2.30}$$

Ainsi, on a les relations suivantes :

$$\boxed{I(X,Y) = H(X) + H(Y) - H(X,Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)} \tag{2.31}$$

L'information mutuelle $I(X,Y)$ mesure la quantité moyenne d'information sur X (ou réduction d'incertitude moyenne) qui résulte de la connaissance de Y.

La figure 2.2 montre graphiquement les relations entre les différentes entropies et l'information mutuelle.

Alors que $i(x_i, y_j)$ peut être négatif, on a $I(X,Y) \geq 0$.

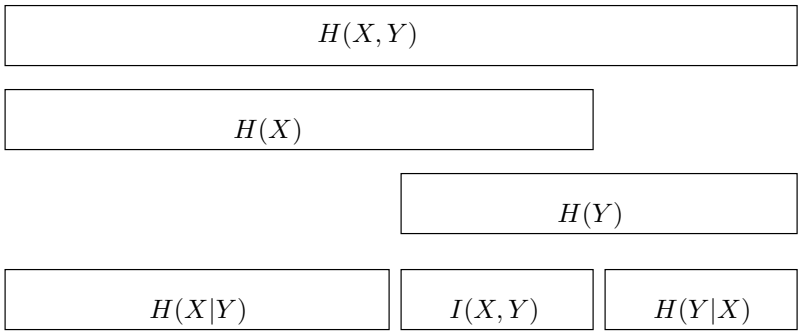


FIG. 2.2 – Relations entre entropie et information mutuelle.

Chapitre 3

Codage de source

3.1 Entropie et Redondance d'une source

Dans le paragraphe précédent, nous avons introduit $H(X)$, la quantité d'information moyenne ou entropie associée à la variable aléatoire discrète X.

Soit une source discrète et stationnaire dont les symboles de sortie sont des symboles Q-aire (la taille de l'alphabet est égale à Q). La sortie de cette source est décrite par la variable aléatoire X. Ainsi l'entropie $H(X)$ est la quantité d'information par symbole moyenne sortant de cette source.

Une source discrète est dite source *sans mémoire* si les symboles en sortie de cette source sont décorrélés. Dans le cas contraire, on dira que cette source est *avec mémoire*.

Si la source est sans mémoire, $H(X)$ s'obtient comme nous l'avons déjà vu par :

$$H(X) = -\sum_{i=1}^{Q} p_i \log_2 p_i \qquad \text{en Sh/symbole} \tag{3.1}$$

L'entropie $H(X)$ est maximale si les symboles sont équiprobables. Pour des symboles Q-aire, l'entropie maximale est égale à $H_{MAX} = \log_2 Q$

Si la source est à mémoire, alors son entropie par symbole $H(X)$ s'obtient comme suit :

$$H(X) = \lim_{J \to \infty} \frac{1}{J} H_J(X) \tag{3.2}$$

où $H_J(X)$ est l'entropie par groupe de J symboles.

La redondance d'une source caractérise la différence qu'il existe entre la quantité d'information que transporte cette source et la quantité d'information que cette source transporterait si tous ses symboles étaient équiprobables et indépendants. On a :

$$R_{red} = 1 - \frac{H(X)}{H_{MAX}(X)} \tag{3.3}$$

R_{red} est compris entre 0 (les symboles de la source sont indépendants et équiprobables) et 1 (l'entropie de la source est nulle).

3.2 Codage de source

D'un point de vue général, l'opération de codage de source consiste à associer à chaque *message* issu de la source un *mot* composé d'un ou plusieurs symboles q-aire en cherchant à réduire au maximum le nombre moyen de ces symboles.

Un *message* signifiera selon le contexte, soit un symbole Q-aire issu de la source, soit un ensemble de J symboles Q-aire.

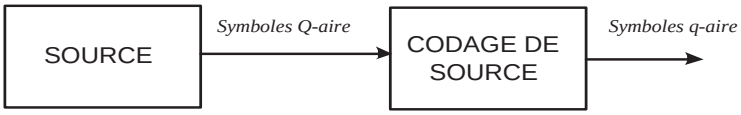


FIG. 3.1 – Codage de source.

Nous nous restreindrons au cas où les symboles de sortie du codeur de source sont des bits ($q = 2$). Cependant la généralisation aux autres alphabets ne présente pas de difficulté.

Le codage de source doit satisfaire les deux critères suivants :

- **décodage unique** : chaque message devra être codé par un mot différent
- **déchiffrabilité** : chaque mot devra pouvoir être dissocié sans ambiguité. Ce critère s'obtient par :
  - codage par mot de longueur fixe
  - codage avec un symbole de séparation distinguable (cas du système morse par exemple)
  - codage par mot de longueur variable mais en évitant qu'un mot ne soit le début d'un autre.

Nous nous intéresserons uniquement au codage par mot de longueur variable qui est la technique de codage la plus efficace lorsque les différents symboles de la source n'ont pas la même probabilité d'apparition.

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

3.3 Codage par mot de longueur variable

exemple 1 : soit une source discrète délivrant les 4 messages a_1, a_2, a_3 et a_4 avec les probabilités respectives $P(a_1) = \frac{1}{2}$, $P(a_2) = \frac{1}{4}$ et $P(a_3) = P(a_4) = \frac{1}{8}$.

Un mot est associé à chaque message comme décrit dans la table 10.1.

TAB. 3.1 – code à longueur variable de l'exemple 1

message	**mot**
a_1	1
a_2	00
a_3	01
a_4	10

On peut vérifier que ce codage de source n'est pas décodable instantanément (le mot associé à a_1 est le début du mot associé à a_4). Par exemple, il n'est pas possible de décoder instantanément le message $a_1, a_2, a_1, \ldots$ encodé par la suite 1001 ; il n'est pas possible de savoir si le message émis était a_1, a_2, a_1 ou a_4, a_3.

exemple 2 :

Un mot est associé à chaque message comme décrit dans la table 3.2.

Ce codage de source est instantané.

Sur la figure 3.2, nous présentons l'arbre associé à ce codage de source.

3.4 Inégalité de Kraft

théorème : un code instantané composé de M mots binaires de longueur respective $\{n_1, n_2, \ldots, n_M\}$ avec $n_1 \leq n_2 \leq \cdots \leq n_M$ doit satisfaire l'inégalité suivante :

TAB. 3.2 – code à longueur variable de l'exemple 2

message	**mot**
a_1	0
a_2	10
a_3	110
a_4	111

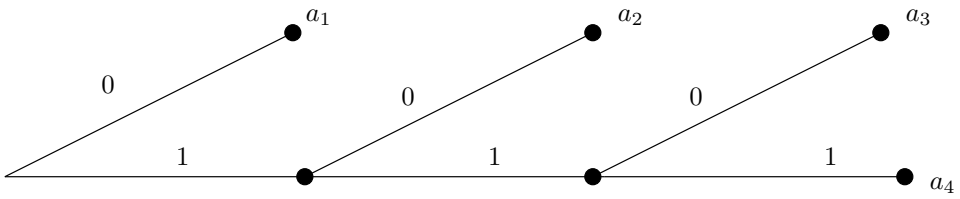


FIG. 3.2 – arbre associé au code de l'exemple 2

$$\sum_{i=1}^{M} 2^{-n_i} \leq 1 \qquad (3.4)$$

démonstration

Un code instantané peut être représenté graphiquement par un arbre binaire complet de profondeur n_M. Chaque feuille du graphe correspond à un des messages de la source. Le mot de code est la séquence de labels du chemin allant de la racine à la feuille.

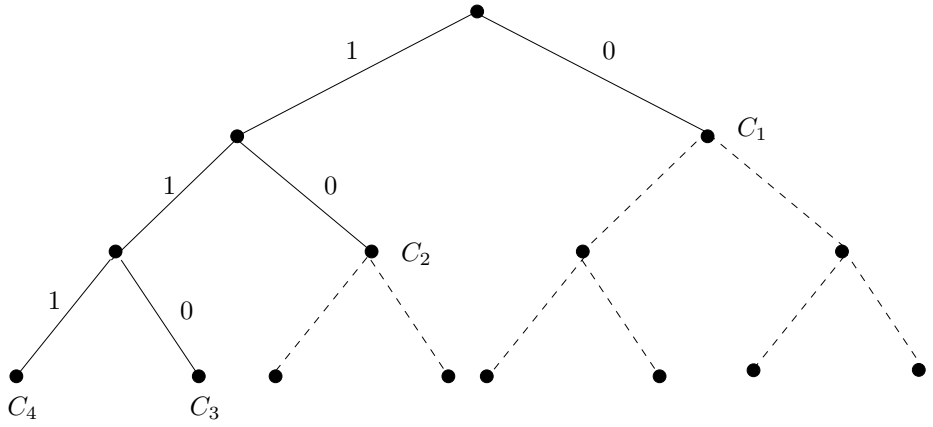


FIG. 3.3 – inégalité de Kraft

Choisissons un nœud de degré n_1 comme premier mot C_1 ; ce choix élimine $2^{n_M - n_1}$ nœuds. Parmi les nœuds restants, on choisit un nœud de degré n_2 comme second mot. Ce choix élimine $2^{n_M - n_2}$ nœuds. On continue cette procédure jusqu'au dernier mot. La condition pour obtenir un codage instantané devient donc :

$$\sum_{i=1}^{M} 2^{n_M - n_i} \leq 2^{n_M}$$

en divisant les deux termes par 2^{n_M}, on obtient bien l'inégalité de Kraft.

3.5 Théorème fondamental du codage de source

Considérons tout d'abord une source sans mémoire d'entropie par symbole $H(X)$. Nous allons démontrer que pour cette source il est possible de construire un code instantané dont la longueur moyenne des mots R_{moy} satisfait l'inégalité suivante :

$$H(X) \leq R_{moy} < H(X) + 1 \tag{3.5}$$

démonstration
On choisit n_i longueur du mot associé au i-ième message comme suit ;

$$n_i = \lceil -\log_2 p_i \rceil \tag{3.6}$$

$\lceil x \rceil$ signifie entier supérieur ou égal à x.
Vérifions qu'un tel code est instantané c'est-à-dire qu'il satisfait l'inégalité de Kraft :

$$\sum_{i=1}^{M} 2^{-n_i} = \sum_{i=1}^{M} 2^{-\lceil -\log_2 p_i \rceil} \leq \sum_{i=1}^{M} 2^{\log_2 p_i} = \sum_{i=1}^{M} p_i = 1 \tag{3.7}$$

Ainsi on a donc :

$$R_{moy} = \sum_{i=1}^{M} p_i n_i = \sum_{i=1}^{M} p_i \lceil -\log_2 p_i \rceil \leq \sum_{i=1}^{M} p_i(-\log_2 p_i + 1) = H(X) + 1 \tag{3.8}$$

Le théorème fondamental du codage de source s'exprime ainsi :

Théorème : pour toute source stationnaire d'entropie par symbole $H(X)$, il existe un procédé de codage de source binaire dont la longueur moyenne R_{moy} des mots est aussi voisine que l'on veut de $H(X)$.

$$\boxed{H(X) \leq R_{moy} < H(X) + \epsilon} \tag{3.9}$$

Considérons une source sans mémoire d'entropie par symbole $H(X)$. En groupant les symboles de cette source par paquet de J symboles, on obtient une nouvelle source. Il est encore possible d'encoder cette source avec un code instantané. La longueur moyenne des mots de ce code R_{Jmoy} satisfait l'inégalité suivante :

$$JH(X) \leq R_{Jmoy} < JH(X) + 1 \tag{3.10}$$

En divisant les différents termes par J on obtient :

$$H(X) \leq R_{moy} < H(X) + \frac{1}{J} \tag{3.11}$$

avec R_{moy} nombre de bits moyens relatifs à un symbole de la source $R_{moy} = \frac{R_{Jmoy}}{J}$ En augmentant J, R_{moy} peut s'approcher asymptotiquement de $H(X)$:

$$H(X) \leq R_{moy} < H(X) + \epsilon \tag{3.12}$$

Ce résultat se généralise immédiatement au cas des sources avec mémoire.

3.6 Débit d'entropie

Soit une source discrète et stationnaire X d'entropie $H(X)$ Sh/symbole délivrant D_S symboles par seconde. On définit le débit d'entropie ou débit informationnel moyen D_I comme suit :

$$D_I = H(X)D_S \qquad \text{en Shannon/seconde} \tag{3.13}$$

Le débit binaire en sortie du codeur D'_B est le produit du débit symbole D_S par le nombre de bits moyens par symbole R_{moy} :

$$D'_B = D_S.R_{moy} \qquad \text{en bit/seconde} \tag{3.14}$$

En sortie du codeur de source binaire, on définit $H'(X)$ l'entropie par bit avec

$$H'(X) = \frac{H(X)}{R_{moy}} \qquad \text{en Shannon/bit} \tag{3.15}$$

Le débit d'entropie D'_I en sortie du codeur s'obtient alors comme précédemment :

$$D'_I = H'(X).D'_B = D_I \quad \text{en Shannon/seconde} \tag{3.16}$$

Comme nous pouvions nous y attendre, le débit d'entropie n'est pas modifié par l'opération de codage de source. D'après le théorème du codage de source, on a :

$$R_{moy} \geq H(X) \tag{3.17}$$

En multipliant les 2 termes par D_S, on obtient :

$$D'_B \geq D_S.H(X) = D_I \tag{3.18}$$

Ainsi, D_I le débit d'entropie de la source est la limite inférieure du débit binaire obtenu après codage de source. En cas d'égalité, on retrouve bien qu'un élément binaire est porteur d'une quantité d'information égale à 1 Shannon. Si la redondance de la séquence en sortie du codeur de source n'est pas nulle, alors un élément binaire sera porteur d'une quantité d'information inférieure à 1 Shannon.

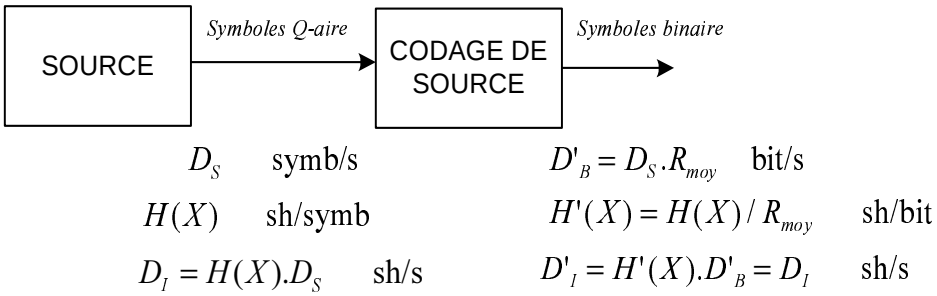


FIG. 3.4 – Débit d'entropie.

3.7 Algorithme d'Huffman

Huffman en 1952 [16] a proposé un algorithme de codage à longueur variable d'une source à L messages. Cet algorithme permet d'obtenir le nombre moyen de bits par mot minimum tout en garantissant un code uniquement décodable et instantané.

On commence par classer la liste des messages de haut en bas par ordre de probabilité décroissante (chaque message correspond à un nœud).

- 1. Choix des deux messages de probabilités moindres.
- 2. Ces deux probabilités sont reliées avec la branche supérieure labelisée à 0 et la branche inférieure labelisée à 1.
- 3. La somme de ces deux probabilités est alors associée au nouveau nœud.
- 4. Suppression des deux messages précédemment choisis puis retour à la phase 1.

On répète cette procédure jusqu'à ce qu'il ne reste plus aucun message. L'arbre ainsi obtenu décrit graphiquement l'ensemble du code. Les mots sont lus en parcourant chaque chemin de la droite vers la gauche.

exemple 3 : soit une source discrète à 8 messages $a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8$ avec les probabilités d'apparition respectives $\{0.16; 0.15; 0.01; 0.05; 0.26; 0.11; 0.14; 0.12\}$ d'entropie $H(X)$=2.7358.

Le codage de Huffman de cette source est donné sur la figure 3.5.

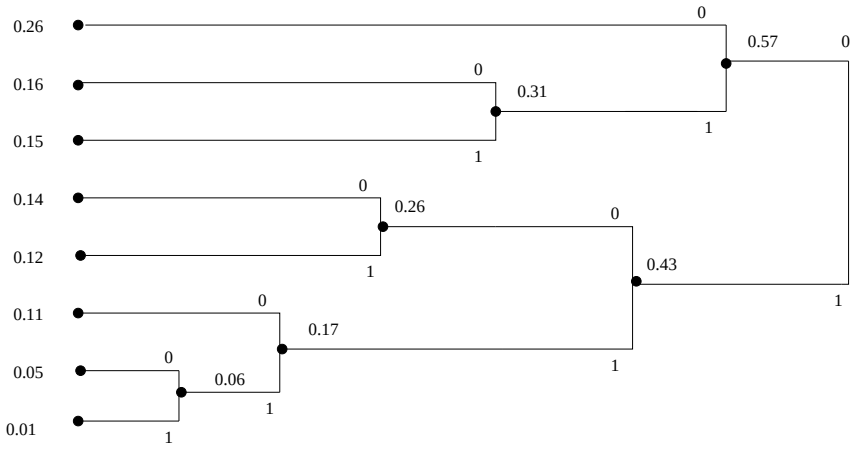


FIG. 3.5 – exemple de codage de Huffman

La table de codage est présentée dans la table 3.3. Le nombre de bit moyen par mot est égal à 2.8.

TAB. 3.3 – table de codage

message	mot	n_i
a_5	00	2
a_1	010	3
a_2	011	3
a_7	100	3
a_8	101	3
a_6	110	3
a_4	1110	4
a_3	1111	4

Pour les sources sans mémoire l'algorithme de Huffman fournit un codage de source optimal. Cependant, lorsque les symboles successifs sont corrélés, la complexité du codage de source utilisant l'algorithme de Huffman devient très grande (le nombre de code est égal à Q^J si les messages sont composés de J symboles Q-aire).

3.8 Etude de l'entropie d'un texte écrit

Dans cette étude, nous réduirons la taille de l'alphabet aux 26 lettres de l'alphabet plus la virgule, le point, le guillemet et espace soit une taille de l'alphabet égale à 30. Les probabilités d'apparition des caractères dans un texte littéraire français sont donnés dans la table 3.4.

TAB. 3.4 – probabilité d'apparition des caractères

a_i	p_i	a_i	p_i
a	0.0734	p	0.0171
b	0.0094	q	0.0059
c	0.0248	r	0.0510
d	0.0325	s	0.0703
e	0.1263	t	0.0533
f	0.0097	x	0.0043
g	0.0091	v	0.0120
h	0.0080	w	0
i	0.0617	x	0.0043
j	0.0035	y	0.0018
k	0.0005	z	0.0005
l	0.0513		0.1717
m	0.0205	'	0.0096
n	0.0504	,	0.0180
o	0.0387	.	0.0086

Si la probabilité d'apparition de chacun des caractères était égale, l'entropie par caractère serait égale à $\log_2 30 = 4,9$ Shannon/caractère.

L'application de (3.1) donne une entropie égale à 4.09 Shannon/caractère. Ce calcul ne tient pas compte de la corrélation entre les caractères successifs. Pour montrer un aperçu de la dépendance entre les caractères successifs, sur un texte littéraire en langue française, nous avons groupé les caractères par doublet et représenté sur la figure 3.8 la probabilité d'apparition de chacun de ces doublets. Les caractères associés aux colonnes et aux lignes correspondent respectivement au premier et au dernier caractère des doublets. Par exemple, on peut voir que le doublet $\{qu\}$ a une forte probabilité d'apparition alors que tous les autres doublets commençant par la lettre q ont une probabilité d'apparition nulle.

En groupant les caractères 2 par 2, on obtient une entropie par caractère de 3,6 Shannon/caractère soit sensiblement inférieure à l'entropie par caractère précédente.

Différentes études on montré que pour un texte littéraire, l'entropie est encore bien plus faible : de l'ordre de 1 Shannon/caractère.

Pour mettre encore en évidence la redondance de la langue française prenons l'exemple suivant [20] [8] : l'objectif est pour le candidat de déterminer lettre après lettre une phrase qu'il ignore. Après avoir déterminé correctement une lettre, on note le nombre de tentatives qui ont été nécessaire pour déterminer celle-ci. Le candidat passe ensuite à la lettre suivante. Voici deux résultats obtenus avec la phrase suivante :

	L	E	S		V	A	C	A	N	C	E	S		S	O	N	T		T	E	R	M	I	N	E	E	S
candidat 1 :	1	1	2	1	18	2	1	2	1	1	1	1	1	2	1	1	1	1	3	1	1	1	1	1	1	1	1
candidat 2 :	1	1	1	1	13	4	4	5	1	1	1	1	1	4	1	1	1	1	9	1	1	1	1	1	1	1	1

Il est à noter que dans beaucoup de cas, les candidats déterminent la lettre suivante dès la première tentative. Excepté au début des mots et des syllabes, les autres lettres sont déterminées très aisément. On peut imaginer un codage de source très efficace utilisant ces propriétés : si au

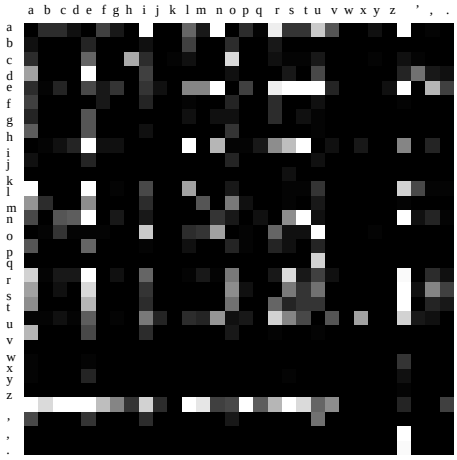


FIG. 3.6 – probabilité d'apparition des caractères

lieu de coder successivement les symboles, on code le nombre de tentatives, on voit bien qu'il sera possible de réduire fortement le nombre de bits nécessaire pour transmettre cette phrase. Ceci implique qu'au décodage nous effectuerons la procédure inverse en utilisant des tables de décodage très complexes. Ce système bien que peu réaliste, nous permet d'illustrer les principes utilisés par le codage arithmétique (codage non traité dans ce document bien que très performant [20] [8] [1] [26]) et par l'algorithme de Lempel-Ziv [32]. C'est ce dernier que nous allons maintenant présenter.

3.9 Algorithme de Lempel-Ziv

Cet algorithme, proposé en 1978 est indépendant des propriétés statistiques de la source. L'algorithme de Lempel-Ziv utilise pour coder une liste de suites stockées dans un dictionnaire.

Le principe de l'algorithme de Lempel-Ziv consiste à découper la séquence de sortie de la source en petites suites de longueurs variables. Les suites qui sont stockées dans un dictionnaire initialement vide sont appelées les suites prototypes. Une suite nouvelle est ajoutée dans le dictionnaire chaque fois qu'elle est différente des suites prototypes déjà stockées. De plus, cette suite à laquelle on ajoute un bit 0 ou 1 ne doit pas être déjà présente dans le dictionnaire.

exemple : considérons le début de séquence binaire issue d'une source :

0010000011000100100000101100010000010000110001010100001000001100000101100000011

Cette séquence est décomposée en suite comme suit :

0, **01**, 00, 000, **1**, 10, **001**, **0010**, **0000**, **101**, **100**, **010**, 00001, **000011**, **0001**, **0101**, **000010**, **0000110**, **0000101**, **1000**, **00011**

Les suites prototypes en gras correspondent aux 16 suites prototypes stockées dans le dictionnaire (la suite 00001 n'est pas stockée dans le dictionnaire car les suites **000010** et **000011** sont déjà présentes dans ce dictionnaire). La table 3.5 donne la liste des 16 suites prototypes dans cet exemple. Chaque suite prototype est ici codée avec un mot de 4 bits.

L'arbre des suites prototypes stockées est présenté sur la figure 3.7.

Finalement, la séquence binaire issue d'une source est décomposée en utilisant les suites prototypes stockées dans le dictionnaire :

TAB. 3.5 – liste des suites prototypes

position	**suite prototype**	**mot de code**
1	1	0000
2	01	0001
3	001	0010
4	010	0011
5	100	0100
6	101	0101
7	0000	0110
8	0001	0111
9	0010	1000
10	0101	1001
11	1000	1010
12	00011	1011
13	000010	1100
14	000011	1101
15	0000101	1110
16	0000110	1111

0010, 0000110, 0010, 010, 0000101, 1000, 1000, 0010, 00011, 0001, 0101, 000010, 0000110, 0000101, 1000, 00011

La sortie du codeur de source est la suivante :

1000 1111 1000 0011 1110 1010 1010 1000 1011 0111 1001 1101 1111 1110 1010 1011

Le décodeur de source utilisant le même algorithme pourra reconstruire le dictionnaire utilisé au codage et donc reconstituer instantanément la séquence émise par la source.

Il est à noter que le codage Lempel-Ziv encode les 79 bits de la séquence de la source en 16 mots de 4 bits soit 64 bits au total. Malgré la courte longueur de la séquence, l'algorithme apporte déjà une sensible réduction du nombre de bits. En pratique, le contenu du dictionnaire est adapté dynamiquement en fonction de l'évolution des caractéristiques de la source.

L'algorithme Lempel-Ziv et ses variantes sont utilisés pour la compression des fichiers informatiques.

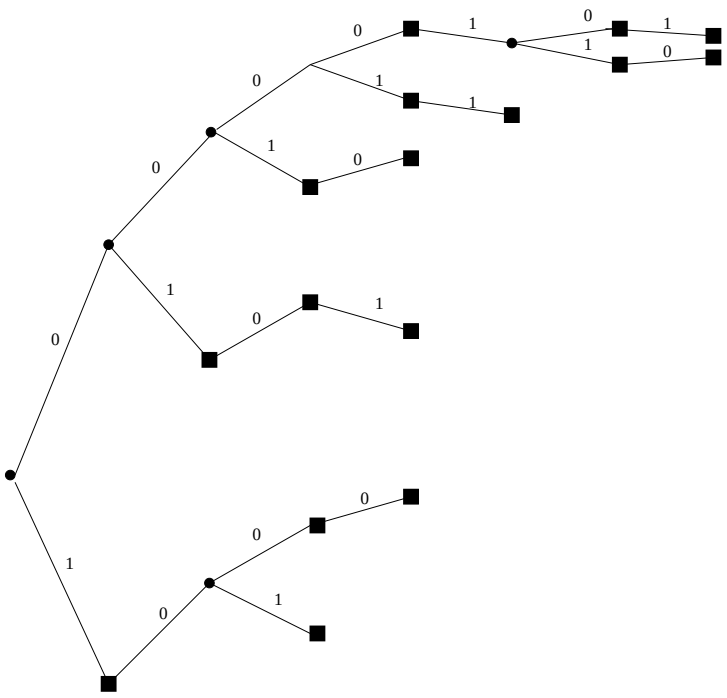


FIG. 3.7 – arbre représentant les suites prototypes

Chapitre 4

Codage pour les sources analogiques

4.1 Échantillonnage et Quantification

4.1.1 Rappel sur le théorème de l'échantillonnage

Soit le signal $x(t)$ à bande limitée B issu d'une source analogique. En utilisant le théorème de l'échantillonnage, on montre que ce signal peut être représenté comme suit :

$$x(t) = \sum_{k=\infty}^{+\infty} x(kT)\mathrm{sinc}\Big(\frac{\pi}{T}(t-kT)\Big) \tag{4.1}$$

avec $\mathrm{sinc}(x) = \frac{sin(x)}{x}$.

La suite $x(kT)$ représente les échantillons du signal $x(t)$ aux instants $kT = \frac{k}{2B}$. Cette suite est donc obtenue par échantillonnage de $x(t)$ à la fréquence de $2B$ échantillons par seconde.

Ainsi, la sortie de la source analogique peut être convertie en un signal à temps discret équivalent sans perte d'information. Pour vraiment disposer d'un signal numérique, il reste à quantifier l'amplitude des échantillons sur un nombre de niveaux finis.

4.1.2 Quantification

La quantification consiste à quantifier l'amplitude possible des échantillons sur L valeurs. Lorsque les L valeurs sont régulièrement espacées, on dit que la quantification est uniforme. La valeur $\hat{x}$ choisie est la plus proche au sens de la distance euclidienne de l'amplitude x de l'échantillon. Si L est une puissance de 2, ($L = 2^R$) alors chaque échantillon quantifié $\tilde{x}$ pourra être représenté par un mot binaire de R bits (opération de codage).

$$R = \log_2 L \qquad \text{bits/échantillon} \tag{4.2}$$

définition : soit un ensemble d'intervalles ou cellules $\mathcal{S} = [s_1, s_2, \ldots, s_L]$ et un ensemble de valeurs ou points $\mathcal{Y} = [y_1, y_2, \ldots, y_L]$. L'opération de quantification est définie mathématiquement par la relation suivante :

$$\tilde{x} = y_i \qquad \text{pour} \quad x \in S_i \tag{4.3}$$

Chaque intervalle ou cellule S_i est bornée par deux seuils notés a_{i-1} et a_i. Ainsi, la largeur de S_i est égale à $a_i - a_{i-1}$ La quantification est dite uniforme si tous les intervalles ont la même largeur notée Δ.

Un exemple de quantification uniforme est donné sur la figure 4.1 pour $L = 8$.

La relation entre l'amplitude de l'échantillon x et l'amplitude de l'échantillon après quantification uniforme $\tilde{x}$ est présentée sur la figure 4.2 pour $L = 8$.

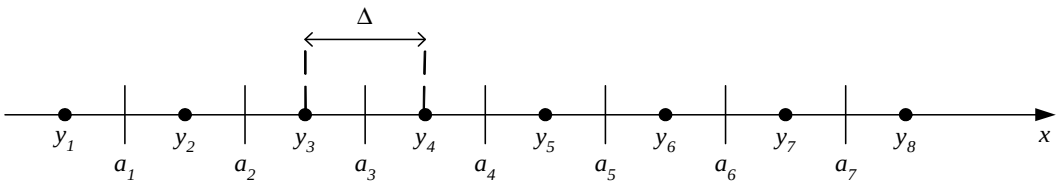


FIG. 4.1 – Quantification uniforme $L = 8$

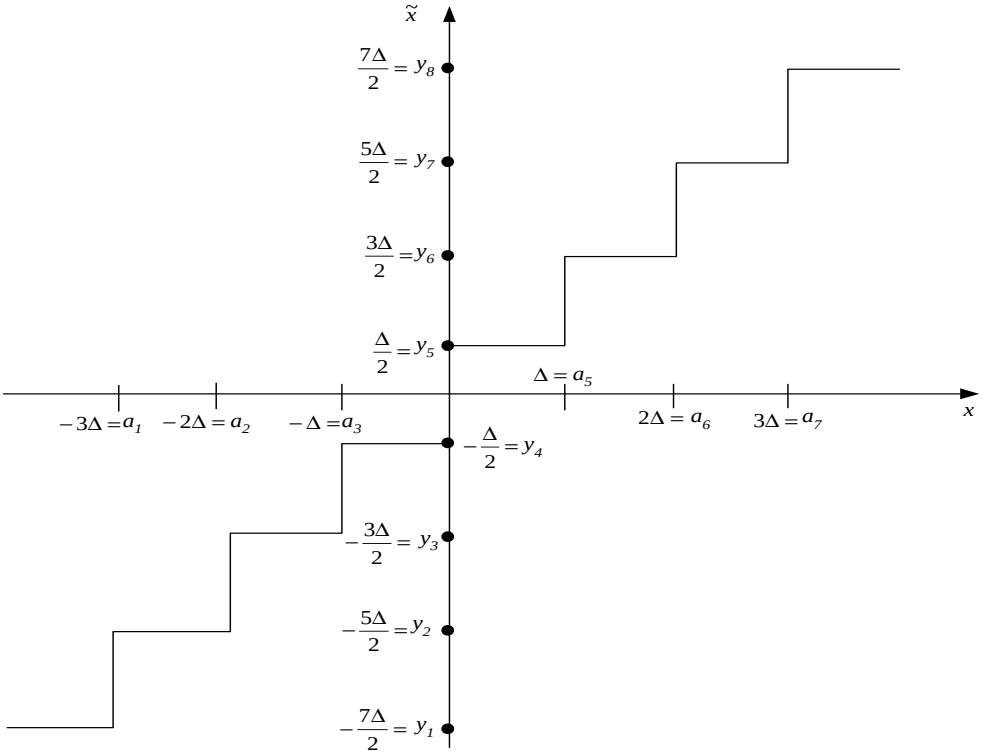


FIG. 4.2 – Quantification uniforme $L = 8$

Un convertisseur analogique numérique (CAN) classique réalise à la fois l'opération de quantification uniforme et de codage binaire. Pour un CAN $R = 8$ bits/échantillon, on a $L = 256$.

Un exemple de quantification non uniforme est donné sur la figure 4.3 pour $L = 8$.

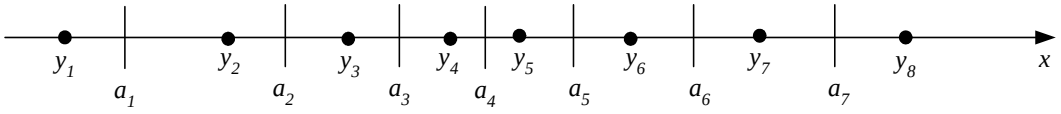


FIG. 4.3 – Quantification non uniforme $L = 8$

La quantification uniforme comme la quantification non uniforme sont des quantifications scalaires : on associe à chaque échantillon un mot binaire. Il est possible de grouper plusieurs échantillons ensemble avant de réaliser l'opération de quantification de ce groupe d'échantillons : on parle alors de quantification vectorielle.

La qualité d'un quantificateur peut être mesurée par la différence entre le signal quantifié et le signal d'origine.

La mesure la plus utilisée est l'erreur quadratique :

$$e(x, \tilde{x}) = (x - \tilde{x})^2 \tag{4.4}$$

définition : à partir de l'erreur quadratique, on peut définir la distorsion moyenne D

$$\begin{aligned} D &= E(e(x, \tilde{x})) \\ &= \int_{-\infty}^{+\infty} e(x, \tilde{x}) f(x) dx \\ &= \sum_i \int_{S_i} e(x, y_i) f(x) dx \end{aligned} \tag{4.5}$$

où $f(x)$ est la densité de probabilité de x.

Ainsi, lorsque la densité de probabilité $f(x)$ est connue, l'objectif de la quantification est de coder les échantillons de la source avec le minimum de bits tout en garantissant la plus petite distorsion moyenne D.

définition : on définit la fonction $R(D)$ (*rate distortion function* en anglais) comme le nombre de bits minimum permettant de représenter une suite d'échantillons avec une distorsion moyenne donnée D :

$$R(D) = minR \qquad \text{en bits} \tag{4.6}$$

On peut montrer que l'inégalité suivante est toujours vraie :

$$R(D) \leq \frac{1}{2} \log_2 \frac{\sigma_x^2}{D} \qquad 0 \leq D \leq \sigma_x^2 \tag{4.7}$$

où σ_x^2 est la variance de la source.

En introduisant la fonction $D(R)$ (*distortion rate function* en anglais), l'expression (4.7) peut aussi s'écrire :

$$D(R) \leq \sigma_x^2 2^{-2R} \tag{4.8}$$

Les deux inégalités ci-dessus se transforment en égalités pour une distribution gaussienne. Il faut noter qu'il s'agit d'une limite théorique qu'une simple quantification uniforme ne permet pas d'atteindre.

Sur la figure 4.4, nous présentons les valeurs optimales y_i avec une quantification uniforme (*) et non uniforme (o) pour $L = 8$ dans le cas d'une source gaussienne.

Dans ce cas simple ($R = 3$ bits/echantillon et $\sigma_x = 1$), les distorsions moyennes des quantifications uniforme et non uniforme sont respectivement égales à -14,27 dB et -14,62 dB. La limite théorique est $10 \log_{10} 2^{-6} = -18.06$ dB . Comme les probabilités de chaque intervalle ne sont pas identiques, on peut appliquer un codage de Huffman après l'opération de quantification pour réduire encore le débit binaire. Ce codage entropique permet de gagner ici environ 2 dB. Pour atteindre la limite théorique, il faudra utiliser une quantification vectorielle bien plus difficile à mettre en oeuvre qu'une simple quantification scalaire !

L'opération de quantification introduit une erreur entre l'amplitude et l'amplitude quantifiée. On a la relation suivante :

$$\tilde{x} = x + q \tag{4.9}$$

Pour la quantification uniforme, l'erreur de quantification q est comprise entre $-\frac{\Delta}{2}$ et $+\frac{\Delta}{2}$.

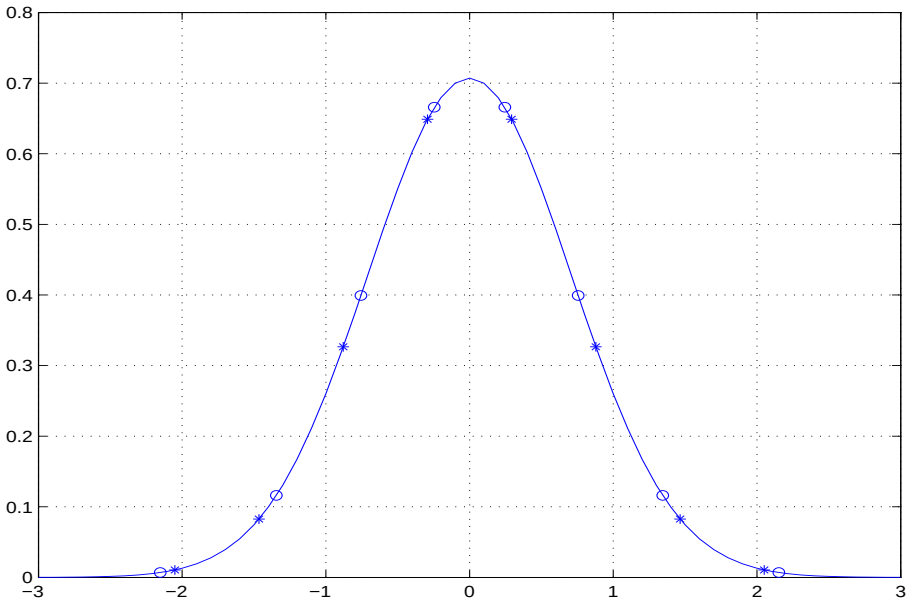


FIG. 4.4 – Quantification uniforme et non uniforme $L = 8$ pour une source gaussienne de variance $\sigma_x = 1$

En considérant que l'amplitude du signal est suffisamment grande devant Δ, la densité de probabilité de q est bien approximativement uniforme. Calculons l'erreur quadratique moyenne $E(q^2)$ (qui est égale ici à la distorsion moyenne D du quantificateur uniforme) :

$$p(q) = \frac{1}{\Delta} \qquad -\frac{\Delta}{2} \leq q \leq \frac{\Delta}{2} \tag{4.10}$$

$$\begin{aligned} E(q^2) &= \int_{-\infty}^{+\infty} q^2 p(q) dq \\ &= \frac{1}{\Delta} \int_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} q^2 dq = \frac{\Delta^2}{12} \end{aligned} \tag{4.11}$$

Si la quantification uniforme est réalisée sur L niveaux avec $R = \log_2 L$ et que la dynamique du signal issu de la source est égal à A avec $A = \Delta L = \Delta 2^R$, alors le rapport signal à bruit en décibel s'exprime comme suit :

$$\begin{aligned} SNR &= 10 \log_{10} \frac{var(x)}{E(q^2)} \\ &= 10 \log_{10} var(x) - 10 \log_{10} \frac{\Delta^2}{12} \\ &= 10 \log_{10} var(x) - 10 \log_{10} \frac{A^2}{12} + 10 \log_{10} 2^{2R} \\ &= 10 \log_{10} var(x) - 10 \log_{10} \frac{A^2}{12} + \frac{\ln 2}{\ln 10} 20R \\ &= 10 \log_{10} var(x) - 10 \log_{10} \frac{A^2}{12} + 6R \end{aligned} \tag{4.12}$$

On peut noter qu'un bit supplémentaire améliore le rapport signal à bruit de 6 dB.

Si on suppose que le signal est une sinusoide d'amplitude crête à crête A (soit une amplitude crête de $A/2$), on a :

$$var(x) = \frac{A^2}{8} \tag{4.13}$$

A partir de l'expression (4.12), on obtient :

$$\begin{aligned} SNR &= 10\log_{10}\frac{3}{2} + 6R \\ &= 1,76 + 6R \quad dB \end{aligned} \tag{4.14}$$

4.2 Modulation par impulsion et codage

La modulation PCM (*pulse coded modulation* en anglais) aussi notée modulation par impulsion et codage (MIC) constitue la plus simple des techniques de codage. Elle comprend deux fonctions distinctes : l'opération de quantification et l'opération de codage [1]

Pour un codeur PCM, après échantillonnage, les échantillons sont simplement quantifiés puis codés. La figure 4.5 présente le synoptique d'un codeur PCM (sans le codeur binaire). Cette technique est bien adaptée aux sources sans mémoire.

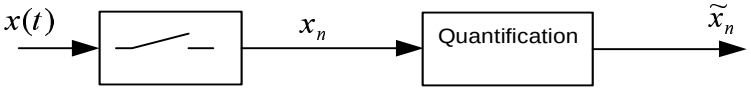


FIG. 4.5 – Codeur PCM

On a la relation suivante entre la sortie du codeur PCM et l'entrée :

$$\tilde{x_n} = x_n + q_n \tag{4.15}$$

où q_n est l'erreur de quantification

Dans de nombreuses applications comme le traitement de la parole par exemple, les signaux de petites amplitudes se produisent plus fréquemment que ceux de grandes amplitudes. Pour prendre en compte cette distribution non uniforme, la quantification uniforme n'est pas la meilleure possible. Plusieurs quantifications non uniformes ont été proposées pour améliorer les performances. En pratique, la quantification non uniforme peut être vue comme la concaténation d'une table de correspondance (*look up table* en anglais) appelée aussi compresseur et d'une quanfication uniforme. Le compresseur réalise l'opération non linéaire. Comme l'idée principale est de diminuer les intervalles pour les petites valeurs de x, les fonctions non linéaires choisies sont des logarithmes. Pour le codage de la parole deux lois de compression sont principalement utilisées :

loi A (système européen)

$$y = \begin{cases} \frac{Ax}{1+\ln A} & \text{si} \quad |x| \leq \frac{1}{A} \quad A = 87,6 \\ (\text{signe de } x)\frac{1+\ln(A|x|)}{1+\ln A} & \text{si} \quad \frac{1}{A} \leq |x| \leq 1 \end{cases} \tag{4.16}$$

Pour la loi A la fonction inverse s'écrit :

[1] le mot "modulation" utilisé ici doit être pris avec précaution. En effet historiquement les techniques de codage pour les sources analogiques comprenaient trois éléments : la quantification, le codage et la modulation. Cependant aujourd'hui seules les deux premières fonctions composent les techniques de codage

$$x = (\text{signe de } y) \begin{cases} \frac{|y|(1+\ln(A))}{A} & \text{si} \quad 0 \leq |y| \leq \frac{1}{1+\ln(A)} \\ \frac{\exp(|y|(1+\ln(A))-1)}{A} & \text{si} \quad \frac{1}{1+\ln(A)} \leq |y| \leq 1 \end{cases} \tag{4.17}$$

loi μ (système américain et canadien)

$$y = (\text{signe de } x)\frac{\ln(1+\mu(x))}{\ln(1+\mu)} \qquad \text{avec} \quad \mu = 255 \quad \text{et} \quad |x| \leq 1 \tag{4.18}$$

Pour la loi μ la fonction inverse s'écrit :

$$x = (\text{signe de } y)\left(\frac{1}{\mu}\right)\left[(1+\mu)^{|y|} - 1\right] \qquad \text{avec} \quad |y| \leq 1 \tag{4.19}$$

Dans les deux lois, les logarithmes sont népériens.

Pour un signal de parole standard, la loi A apporte une réduction de 24 dB du bruit de quantification par rapport à une quantification uniforme.

Sur la figure 4.6, nous présentons les caractéristiques de transfert relative à la loi A et à la loi μ (les courbes sont pratiquement superposées !)

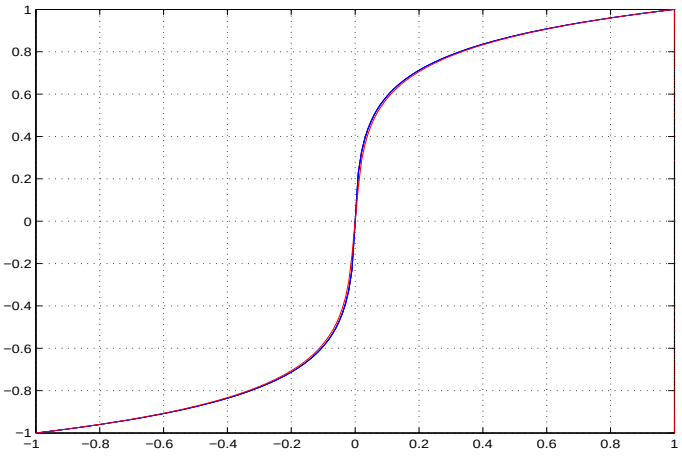


FIG. 4.6 – Caractéristiques de transfert d'un compresseur basé sur la loi A et la loi μ

En pratique, la compression peut être réalisée à partir d'une quantification uniforme sur 12 bits. On modélise la loi par 13 segments. La table d'encodage est présentée ci-dessous :

4.3 Techniques de codage pour les sources analogiques à mémoire

4.3.1 Introduction

Lorsque les échantillons à la sortie de la source sont corrélés (source avec mémoire), il existe 3 grandes familles de techniques pour exploiter cette propriété afin de réduire le nombre de bits nécessaire pour transmettre ces échantillons :

- les techniques basées sur une forme d'onde temporelle comme la modulation Delta, PCM, DPCM, ... souvent utilisé pour le codage de la parole.
- les techniques utilisant une décomposition spectrale comme le codage par sous-bande et le codage par transformée (cosinus discret ou ondelettes).
- les techniques basées sur des modèles de source comme le codage linéaire prédictif (LPC) utilisés pour le codage de la parole à très bas débit.

segment	mot d'entrée 12b												mot de sortie 8b							
13	0	1	X_1	X_2	X_3	X_4	N	N	N	N	N	N	0	1	1	1	X_1	X_2	X_3	X_4
12	0	0	1	X_1	X_2	X_3	X_4	N	N	N	N	N	0	1	1	0	X_1	X_2	X_3	X_4
11	0	0	0	1	X_1	X_2	X_3	X_4	N	N	N	N	0	1	0	1	X_1	X_2	X_3	X_4
10	0	0	0	0	1	X_1	X_2	X_3	X_4	N	N	N	0	1	0	0	X_1	X_2	X_3	X_4
9	0	0	0	0	0	1	X_1	X_2	X_3	X_4	N	N	0	0	1	1	X_1	X_2	X_3	X_4
8	0	0	0	0	0	0	1	X_1	X_2	X_3	X_4	N	0	0	1	0	X_1	X_2	X_3	X_4
7	0	0	0	0	0	0	0	1	X_1	X_2	X_3	X_4	0	0	0	1	X_1	X_2	X_3	X_4
7	0	0	0	0	0	0	0	0	X_1	X_2	X_3	X_4	0	0	0	0	X_1	X_2	X_3	X_4
7	1	0	0	0	0	0	0	0	X_1	X_2	X_3	X_4	1	0	0	0	X_1	X_2	X_3	X_4
7	1	0	0	0	0	0	0	1	X_1	X_2	X_3	X_4	1	0	0	1	X_1	X_2	X_3	X_4
6	1	0	0	0	0	0	1	X_1	X_2	X_3	X_4	N	1	0	1	0	X_1	X_2	X_3	X_4
5	1	0	0	0	0	1	X_1	X_2	X_3	X_4	N	N	1	0	1	1	X_1	X_2	X_3	X_4
4	1	0	0	0	1	X_1	X_2	X_3	X_4	N	N	N	1	1	0	0	X_1	X_2	X_3	X_4
3	1	0	0	1	X_1	X_2	X_3	X_4	N	N	N	N	1	1	0	1	X_1	X_2	X_3	X_4
2	1	0	1	X_1	X_2	X_3	X_4	N	N	N	N	N	1	1	1	0	X_1	X_2	X_3	X_4
1	1	1	X_1	X_2	X_3	X_4	N	N	N	N	N	N	1	1	1	1	X_1	X_2	X_3	X_4

4.3.2 Modulation Delta

Nous savons que les sources analogiques (son et image) possède une forte redondance qu'une simple modulation PCM ne peut exploiter. Cette redondance est directement liée à la corrélation entre échantillons : on parle aussi de source à mémoire. Lorsque la source est à mémoire, la variation de l'amplitude entre les échantillons successifs est relativement faible. Ainsi en quantifiant les différences entre les amplitudes des échantillons successifs, il est possible de réduire le débit en sortie du codeur.

Le principe de base de la modulation Delta consiste à quantifier la différence d'amplitude e_n entre l'échantillon courant x_n et l'échantillon quantifié $\hat{x}_n$.

$$e_n = x_n - \hat{x}_n \tag{4.20}$$

La quantification est uniquement réalisée sur deux niveaux ($\tilde{e}_n = \pm\Delta$).

La figure 4.7 présente le synoptique d'un modulateur Delta.

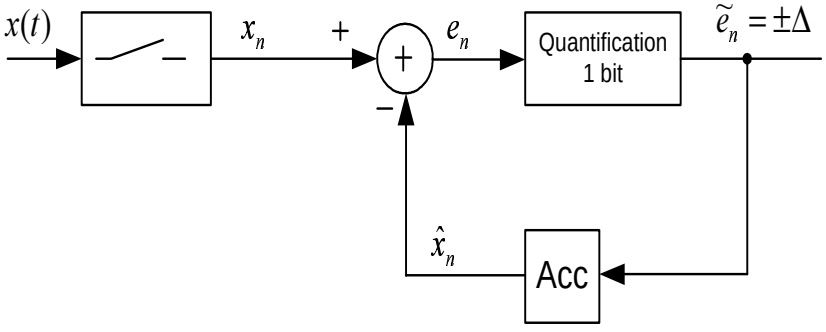


FIG. 4.7 – Modulateur Delta

L'accumulateur réalise l'opération suivante :

$$\hat{x}_n = \hat{x}_{n-1} + \tilde{e}_{n-1} \tag{4.21}$$

Si à l'instant n, on a $x_n > \hat{x}_n$, alors $\tilde{e}_n = +\Delta$ et la sortie de l'accumulateur à l'instant $n+1$ est incrémentée de Δ : $\hat{x}_{n+1} = \hat{x}_n + \Delta$

Si à l'instant n, on a $x_n \leq \hat{x}_n$, alors $\tilde{e}_n = -\Delta$ et la sortie de l'accumulateur à l'instant $n+1$ est décrémentée de Δ : $\hat{x}_{n+1} = \hat{x}_n - \Delta$

Le synoptique de l'accumulateur est présenté sur la figure 4.8.

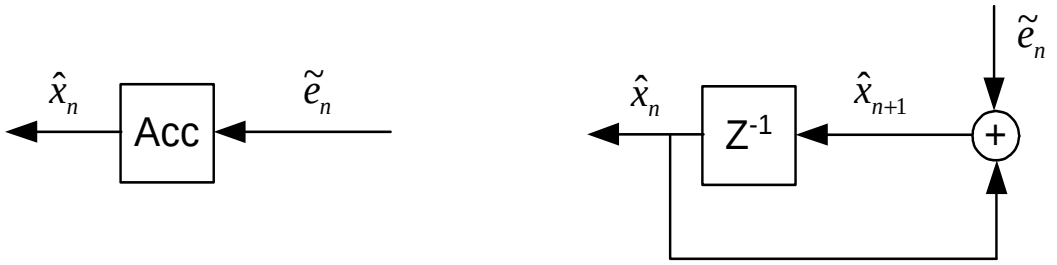


FIG. 4.8 – synoptique de l'accumulateur

Le synoptique du démodulateur Delta est présenté sur la figure 4.9.

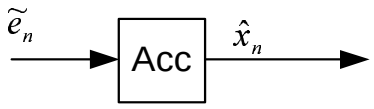


FIG. 4.9 – Démodulateur Delta

L'erreur de quantification q_n apportée par l'opération de quantification est donnée par :

$$q_n = e_n - \tilde{e}_n \tag{4.22}$$

Il est alors possible d'exprimer l'échantillon estimé $\hat{x}_n$ à partir de $\hat{x}_{n-1}$ et de l'erreur de quantification :

$$\begin{aligned} \hat{x}_n &= \hat{x}_{n-1} + \tilde{e}_{n-1} \\ &= (x_{n-1} - q_{n-1} - \tilde{e}_{n-1}) + \tilde{e}_{n-1} \\ &= x_{n-1} - q_{n-1} \end{aligned} \tag{4.23}$$

Ainsi l'échantillon estimé $\hat{x}_n$ est égal à l'échantillon précédent x_{n-1} entaché de l'erreur de quantification q_{n-1}

Un exemple de fonctionnement est donné sur la figure 4.10.

On peut observer deux types d'erreurs : l'erreur de poursuite est liée à la pente de $\hat{x}_n$ limitée à Δ/T_{ech}. Pour diminuer l'erreur de poursuite, la fréquence d'échantillonnage doit être égale à 4 à 5 fois la fréquence minimum d'échantillonnage. L'autre solution consiste à augmenter la valeur de Δ. Le second type d'erreur appelé aussi bruit granulaire se produit même si le signal $x(t)$ est constant. En effet, les échantillons estimés $\hat{x}_n$ oscillent alors entre deux pas (bruit crête à crête de Δ. Une solution consiste alors à diminuer Δ. Le choix de Δ est un compromis entre les deux types d'erreurs. Une autre solution efficace consiste à adapter le pas Δ en fonction des variations du signal. C'est le principe utilisé dans la modulation Delta à pente continuellement variable CVSD (*continuously variable slope delta modulation* en anglais).

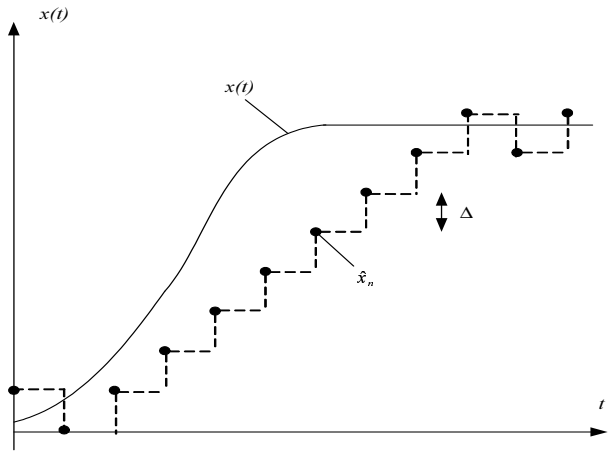


FIG. 4.10 – Exemple d'évolution de $\hat{x}_n$ dans un modulateur Delta

4.3.3 Modulation par impulsion et codage différentiel

Le principe de base de la modulation DPCM (*differential pulse coded modulation* en anglais) aussi notée modulation par impulsion et codage différentiel (MICD) consiste à quantifier la différence d'amplitude e_n entre l'échantillon courant x_n et l'échantillon prédit $\hat{x}_n$. e_n est aussi appelée l'erreur de prédiction.

$$e_n = x_n - \hat{x}_n \tag{4.24}$$

$\hat{x}_n$ est obtenu en utilisant un prédicteur d'ordre P :

$$\hat{x}_n = \sum_{i=1}^{P} a_i x_{n-i} \tag{4.25}$$

Le synoptique du prédicteur d'ordre P est présenté sur la figure 4.11.

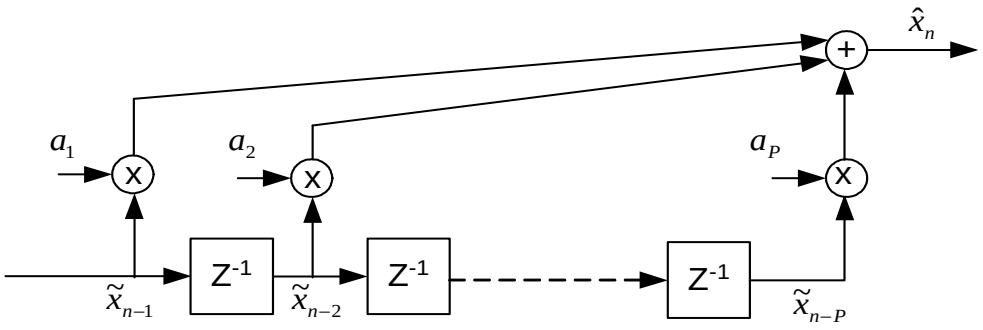


FIG. 4.11 – synoptique du prédicteur d'ordre P

On déterminera les P coefficients de prédiction a_i qui minimisent l'erreur quadratique moyenne $E(e_n^2) = E[(x_n - \hat{x}_n)^2]$. Ces coefficients sont les solutions du système linéaire suivant :

$$\sum_{i=1}^{P} a_i \phi(i-j) = \phi(j) \qquad \text{pour} \quad j = 1, 2, \ldots, P \tag{4.26}$$

où $\phi(m)$ est la fonction d'autocorrélation des échantillons x_n. Ce système se résout efficacement en utilisant l'algorithme de Levinson. Les coefficients a_i sont déterminés en début de transmission mais peuvent aussi être ajustés périodiquement

Avec cette structure, les échantillons à quantifier sont décorrélés et de très faible amplitude et nécessite donc un nombre de bits limités.

La figure 4.12 présente le synoptique d'un codeur DPCM.

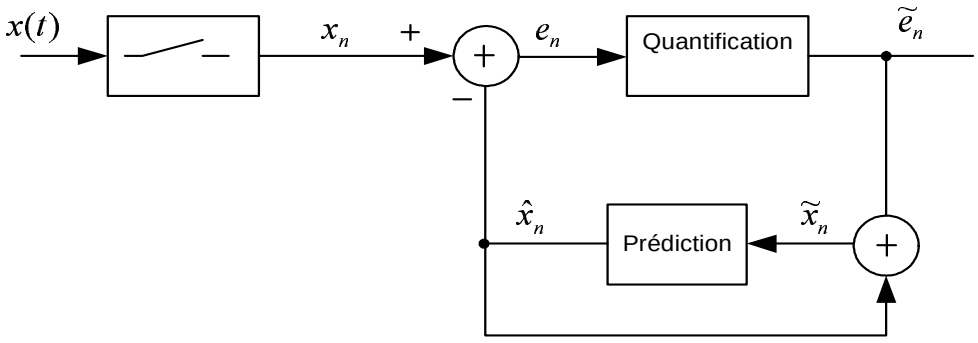


FIG. 4.12 – Codeur DPCM

En entrée du prédicteur, au lieu des échantillons de la source x_n , on utilise les échantillons modifiés par quantification $\tilde{x}_n = \hat{x}_n + \tilde{e}_n$:

$$\hat{x}_n = \sum_{i=1}^{P} a_i \tilde{x}_{n-i} \tag{4.27}$$

On peut vérifier que l'erreur de quantification $q_n = e_n - \tilde{e}_n$ est aussi égale à la différence $x_n - \tilde{x}_n$:

$$\begin{aligned} x_n - \tilde{x}_n &= x_n - \hat{x}_n - \tilde{e}_n \\ &= e_n - \tilde{e}_n \\ &= q_n \end{aligned} \tag{4.28}$$

La figure 4.13 présente le synoptique d'un décodeur DPCM.

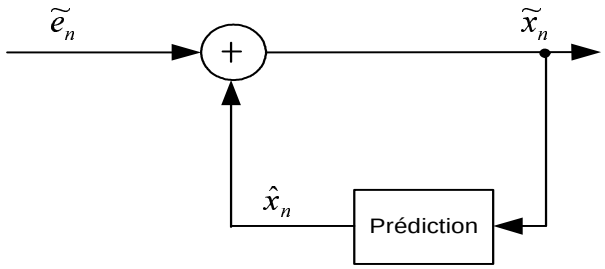


FIG. 4.13 – Décodeur DPCM

Pour le décodage, on utilise exactement le même prédicteur que pour le codage (en considérant l'absence d'erreurs de transmission). Ainsi, on peut reconstruire les échantillons $\tilde{x}_n = \tilde{e}_n + \hat{x}_n$.

Il faut noter que la modulation Delta est une version simplifiée de la modulation DPCM. En effet, pour la modulation Delta la quantification est uniquement réalisée sur un bit et le prédicteur est remplacé par un simple filtre fixe de fonction de transfert Z^{-1}.

Il existe des structures plus complexes de modulateur DPCM utilisant deux filtres de prédiction. Il est également possible d'adapter le pas de quantification en fonction de la variance des échantillons de la source. On parle alors de modulateur DPCM adaptatif (ADPCM).

La modulation DPCM est utilisée pour le codage de la parole dans les standards ITU G.721, G.722, G.723 et G.726.

4.3.4 Codage par décomposition spectrale

4.3.5 Codage basé sur un modèle

4.3.6 Tableau de synthèse

La table 4.1 présente une comparaison des différentes techniques de modulation pour le codage de la parole en considérant une fréquence d'échantillonnage de 8kHz. Les paramètres choisis sont ceux qui sont les plus couramment utilisés.

TAB. 4.1 – Comparaison des modulations pour le codage de la parole

technique	quantificateur	nbr de bits	débit
PCM	uniforme	12 bits	96 kb/s
log PCM	logarithmique	8 bits	64 kb/s
DPCM	logarithmique	4-6 bits	32-48 kb/s
ADPCM	adaptative	3-4 bits	24-32 kb/s
Delta	binaire	1 bit	32-64 kb/s
Delta adaptatif	binaire	1 bit	16-32 kb/s
LPC CELP			2,4-9,6 kb/s

Chapitre 5

Transmission en bande de base

5.1 Introduction

On distingue deux types de transmission numérique : la transmission en bande de base et la transmission par ondes modulées (on dit aussi sur porteuse). Une transmission en bande de base signifie que les symboles à émettre dans le canal de transmission ne subissent pas de translation de leur spectre autour d'une fréquence porteuse. Dans le cas d'une transmission par ondes modulées, les symboles à émettre module une porteuse de fréquence f_c (en amplitude, en phase ou par saut de fréquence). Dans le cas d'une transmission en bande de base et pour la distinguer d'une transmission par ondes modulées, on utilise souvent le terme codeur en ligne à la place de modulateur. Bien que les principes mis en œuvre soient les mêmes, nous étudions d'abord les transmissions en bande de base puis les transmissions par ondes modulées.

D'une manière générale, le signal émis s'écrit de la forme suivante :

$$x(t) = \sum_{k=-\infty}^{\infty} a_k g(t - kT) \tag{5.1}$$

T est la durée de l'impulsion élémentaire ou durée symbole [1]
a_k est le k^{eme} symbole à transmettre
Le choix de l'impulsion $g(t)$ définit les caractéristiques spectrales de la transmission.

5.2 Les codes en ligne

Les critères principaux pour choisir un code en ligne sont les suivants :

1. la densité spectrale du code
2. la densité des transitions dans le signal émis
3. la résistance au bruit

Dans ce paragraphe, nous allons présenter les codes en ligne les plus usuels.

5.2.1 Le code non retour à zéro (NRZ)

Ce code associe un niveau +A à chaque bit égal à "1" et un niveau -A à chaque bit égal à "0". L'impulsion élémentaire $g(t)$ est donc une fonction porte de durée T donnée sur la figure 5.1.

Dans l'annexe 1, nous avons développé le calcul de la densité spectrale $\gamma_{XX}(f)$ d'un signal $x(t)$ émis en bande de base.

Après calcul, on obtient :

[1] Nous utiliserons T pour désigner la durée de l'impulsion élémentaire ou durée symbole. La durée de transmission d'un élément binaire sera notée T_b

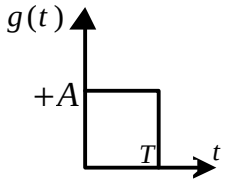


FIG. 5.1 – Impulsion élémentaire $g(t)$.

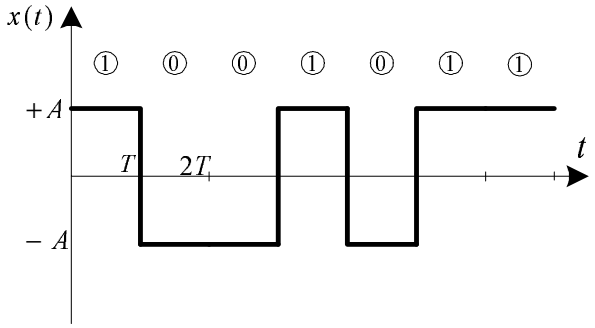


FIG. 5.2 – Signal NRZ.

$$\gamma_{XX}(f) = \frac{1}{T}|G(f)|^2\gamma_{AA}(f) \qquad (5.2)$$

où $|G(f)|^2$ est la densité spectrale de puissance de l'impulsion $g(t)$ et $\gamma_{AA}(f)$ est la transformée de Fourier de la fonction d'autocorrélation $\phi_{aa}(i) = E(a_{k+i}a_k^*)$.

$$\gamma_{AA}(f) = \sum_{i=-\infty}^{+\infty} \phi_{aa}(i)e^{-j2\pi fiT} \qquad (5.3)$$

Ainsi, la densité spectrale de puissance du signal émis est déterminée à partir de la corrélation des symboles et la forme de l'impulsion $g(t)$ (c'est-à-dire du filtre d'émission).

Cette formule est souvent appelée formule de Bennett.

Appliquons cette formule pour le code en ligne NRZ.

Puisque $g(t)$ est une impulsion de largeur T et d'amplitude A, on a :

$$|G(f)|^2 = A^2T^2\left(\frac{\sin(\pi fT)}{\pi fT}\right)^2 \qquad (5.4)$$

Dans le cas du code NRZ, les symboles a_k peuvent prendre les valeurs +1 et -1. Leur moyenne est nulle et leur variance σ^2 est égale à 1. En conséquence, on a $\gamma_{AA}(f) = 1$ (voir le cas particulier traité dans l'annexe 1). Finalement, on obtient donc la densité spectrale de puissance suivante :

$$\gamma_{XX}(f) = A^2T\left(\frac{\sin(\pi fT)}{\pi fT}\right)^2 \qquad (5.5)$$

La densité spectrale de puissance du code NRZ est présentée sur la figure 5.3

L'occupation spectrale de ce code est théoriquement infinie. Cependant, le premier lobe de la densité spectrale de puissance comprend 90% de la puissance du signal (0 à $1/T$). La densité spectrale de puissance s'annule à toutes les fréquences $1/T$.

L'absence de raie à la fréquence $1/T$ ne permet pas d'extraire directement la fréquence rythme à la réception. En effet, les transitions sur le signal émis sont les mêmes que celles de la séquence

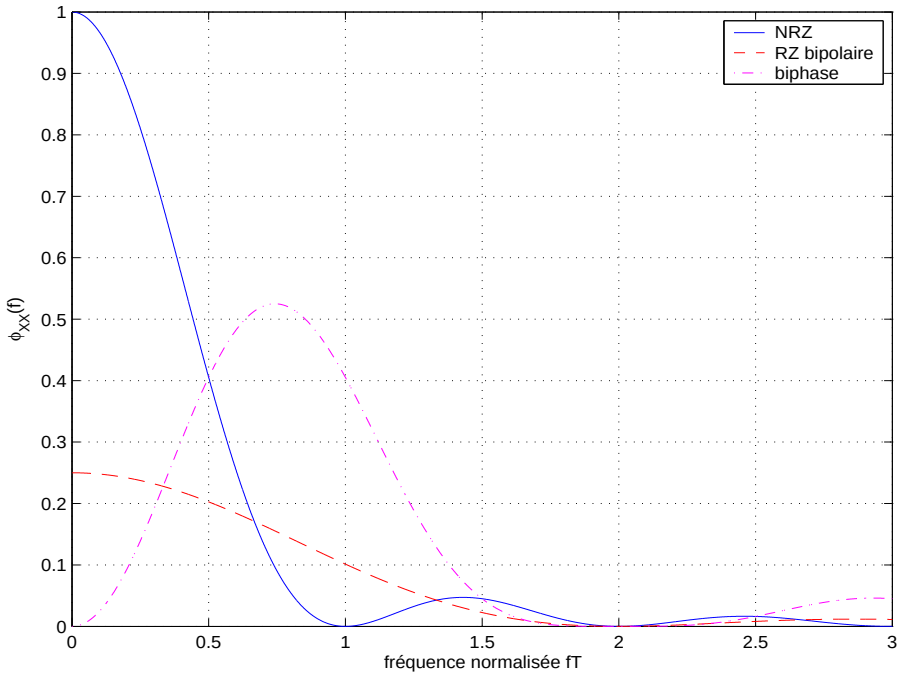


FIG. 5.3 – Densité spectrale des codes NRZ, RZ bipolaire et biphase.

binaire. Ainsi lors d'une longue suite de 0 ou de 1, on perd toute information de rythme. En pratique, pour pallier à cette difficulté, on ajoute périodiquement une séquence pilote composée de bits connus (séquence avec beaucoup de transition).

5.2.2 Code retour à zéro (RZ) unipolaire

Ce code associe à chaque bit égal à "1" un niveau +A pendant une durée $T/2$ puis un niveau O pendant $T/2$. A chaque bit égal à "0" est associé un niveau 0.

Ce code est aussi appelé code RZ 1/2.

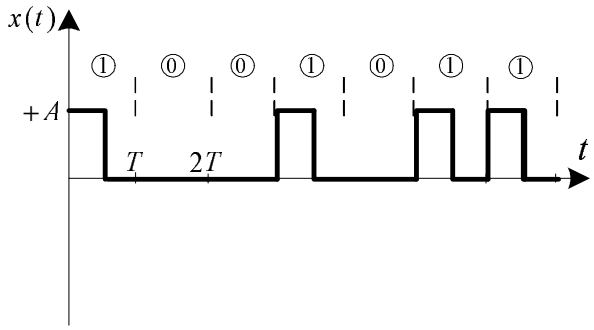


FIG. 5.4 – Signal RZ unipolaire.

La moyenne μ de la séquence est égale à 1/2 et la variance centrée σ^2 est égale à 1/4. Ainsi, on a :

$$\gamma_{AA}(f) = \frac{1}{4} + \frac{1}{4T} \sum_{i=-\infty}^{+\infty} \delta\left(f - \frac{i}{T}\right) \tag{5.6}$$

et

$$G(f) = \frac{AT}{2}\frac{\sin(\pi fT/2)}{\pi fT/2} \tag{5.7}$$

En utilisant les résultats du cas particulier 2 de l'annexe, la densité spectrale de puissance du code RZ unipolaire est la suivante :

$$\gamma_{XX}(f) = \frac{A^2T}{16}\left(\frac{\sin(\pi fT/2)}{\pi fT/2}\right)^2 + \frac{A^2}{16}\sum_{i=-\infty}^{+\infty}\delta\Big(f-\frac{i}{T}\Big)\left(\frac{\sin(\pi fT/2)}{\pi fT/2}\right)^2 \tag{5.8}$$

La raie à la fréquence $1/T$ permet la récupération de rythme par filtrage.

5.2.3 Code retour à zéro (RZ) bipolaire simple

Ce code associe à chaque bit égal à "1" un niveau +A pendant une durée $T/2$ puis un niveau O pendant $T/2$. A chaque bit égal à "0" est associé un niveau -A pendant une durée $T/2$ puis un niveau O pendant $T/2$.

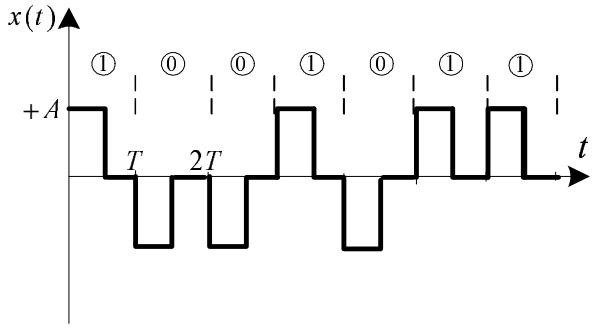


FIG. 5.5 – Signal RZ bipolaire.

Par rapport au code RZ unipolaire, la moyenne de la séquence de symbole est nulle. En conséquence, la densité spectrale ne comporte plus de spectre de raies. On a $\sigma^2 = 1$ et donc

$$\gamma_{XX}(f) = \frac{A^2T}{4}\left(\frac{\sin(\pi fT/2)}{\pi fT/2}\right)^2 \tag{5.9}$$

La densité spectrale de puissance du code RZ bipolaire simple est présentée sur la figure 5.3

5.2.4 Code biphase ou Manchester

Ce code associe à chaque bit égal à "1" un niveau +A pendant une durée $T/2$ puis un niveau -A pendant $T/2$. A chaque bit égal à "0" on associe un niveau -A pendant $T/2$ puis un niveau +A pendant $T/2$.

La densité spectrale de puissance de ce code est donnée par l'expression suivante :

$$\gamma_{XX}(f) = A^2T\frac{(\sin(\pi fT/2))^4}{(\pi fT/2)^2} \tag{5.10}$$

La densité spectrale de puissance du code biphase est présentée sur la figure 5.3

Les transitions régulières permettent d'extraire simplement l'horloge de synchronisation. Cependant, l'occupation spectrale se trouve augmentée par rapport au code NRZ.

Ce code est utilisé pour Ethernet.

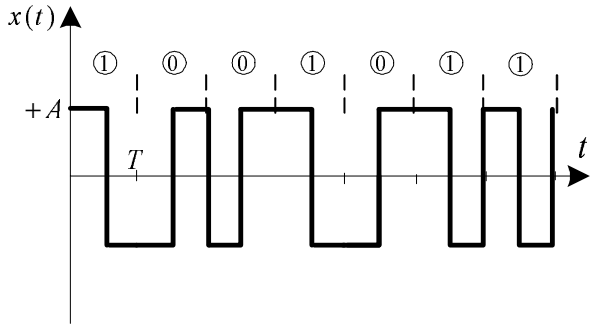


FIG. 5.6 – Signal biphase.

5.2.5 Code bipolaire ou AMI

Ce code associe à chaque bit égal à "1" successivement un niveau +A (signal s_2) et un niveau -A (signal s_3). A chaque bit égal à "0" est associé un niveau 0 (signal s_1).

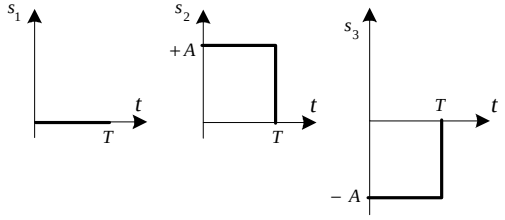


FIG. 5.7 – Signaux de base du code bipolaire

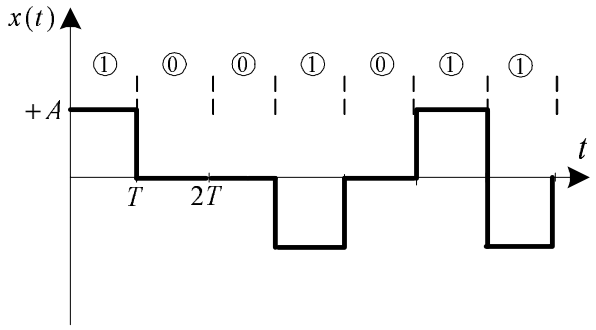


FIG. 5.8 – Signal bipolaire.

Le code bipolaire est aussi appelé code Alternated Mark Inversion (AMI).

Ce code peut aussi être décrit en utilisant un diagramme de transition comportant 2 états distincts $\mathcal{S}_0$ et $\mathcal{S}_1$ comme indiqué sur la figure 5.9.

La densité spectrale de puissance avec un codage bipolaire est donnée par l'expression suivante :

$$\gamma_{XX}(f) = A^2 T \sin^2(\pi f T) \left(\frac{\sin(\pi f T)}{\pi f T} \right)^2 \tag{5.11}$$

La densité spectrale de puissance du code bipolaire est présentée sur la figure 5.10

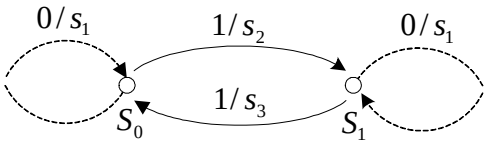


FIG. 5.9 – Diagramme de transition du code bipolaire

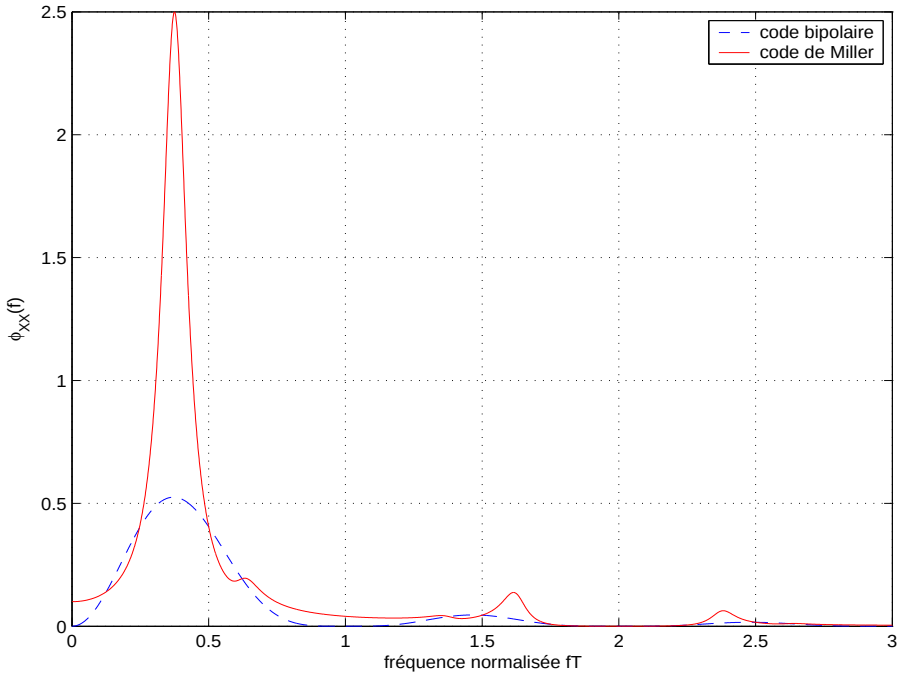


FIG. 5.10 – Densité spectrale des codes bipolaire et de Miller.

Le spectre de ce code ne contient pas de basses fréquences mais occupe une bande plus large que celle du code NRZ. Il peut être utilisé lorsque le canal de transmission ne laisse pas passer les basses fréquences. Le code bipolaire est utilisé dans les raccordements de base du RNIS.

5.2.6 Code de Miller

Ce code associe à chaque bit égal à "1" soit un niveau +A pendant $T/2$ puis un niveau -A (signal s_2) soit un niveau -A pendant $T/2$ puis un niveau +A (signal s_3). A chaque bit égal à "0" on associe un niveau -A (signal s_4) ou un niveau +A (signal s_1). La polarité du signal associé à un bit égal à "1" est choisie de façon à garantir une continuité avec l'impulsion précédente. La polarité du signal associé à un bit égal à "0" est choisie de façon à garantir une continuité avec l'impulsion précédente si celle-ci portait un bit égal à "1".

Ce code peut aussi être décrit en utilisant un diagramme de transition comportant 4 états distincts comme indiqué sur la figure 5.13 [24] .

La densité spectrale de puissance est la suivante [24] :

$$\gamma_{XX}(f) = \frac{1}{2\psi^2(17+8\cos 8\psi)}(23 - 2\cos\psi - 22\cos 2\psi - 12\cos 3\psi + 5\cos 4\psi + 12\cos 5\psi + 2\cos 6\psi - 8\cos 7\psi + 2\cos 8\psi)$$

avec $\psi = \pi f T$

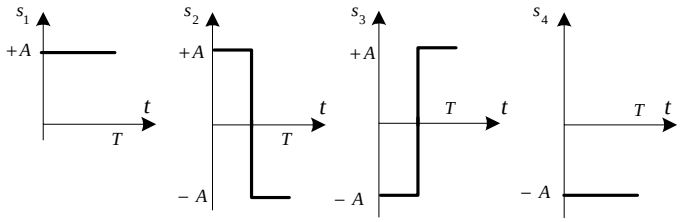


FIG. 5.11 – Signaux de base du code Miller

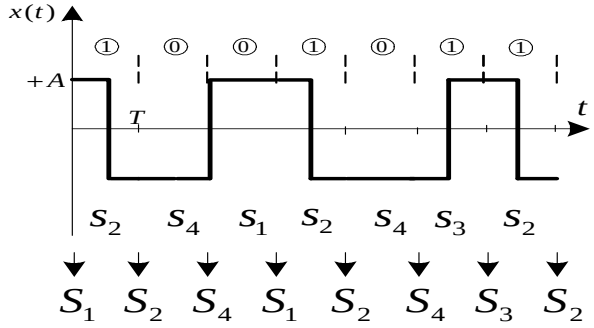


FIG. 5.12 – Signal correspondant au codage de Miller.

La densité spectrale de puissance du code de Miller est présentée sur la figure 5.10

On peut observer que le spectre est étroit autour de la fréquence $0.375/T$ et ne contient pas de basses fréquences. Ces deux propriétés sont interessantes lorsque le canal de transmission ne laisse pas passer les basses fréquences. Ceci justifie l'utilisation de ce code pour l'enregistrement magnétique.

5.2.7 Code NRZ M-aire

Les codes que nous venons d'étudier sont tous des codes binaires. Les applications majeures des codes binaires en ligne restent les transmissions en bande de base sur paires torsadées, cables coaxiaux et fibres optiques. Dans ces applications, l'encombrement spectrale n'est pas un paramètre trop important.

Nous allons maintenant étudier le code NRZ M-aire qui permet de transporter plusieurs bits

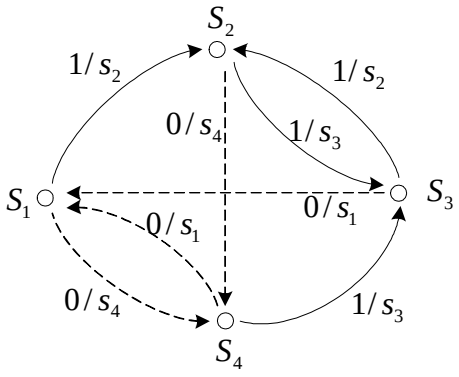


FIG. 5.13 – Diagramme de transition du code Miller

par symbole et donc d'améliorer l'efficacité spectrale du code. Ce code est également à la base des modulations numériques en quadrature souvent utilisées pour les transmissions par ondes modulées.

Comme pour le code NRZ, l'impulsion élémentaire d'un code NRZ M-aire $g(t)$ a une durée T et une amplitude $+A$. Pour un code NRZ M-aire où M est une puissance de 2, les bits sont groupés par paquet de $log_2 M$ bits. Les symboles sont codés suivant l'alphabet $\{\pm 1, \pm 3, \ldots, \pm(M-1)\}$. Ainsi, les symboles peuvent prendre M valeurs différentes.

On a la relation suivante entre T_b la durée de transmission d'un bit et T la durée de transmission d'un symbole :

$$T = T_b log_2 M \tag{5.12}$$

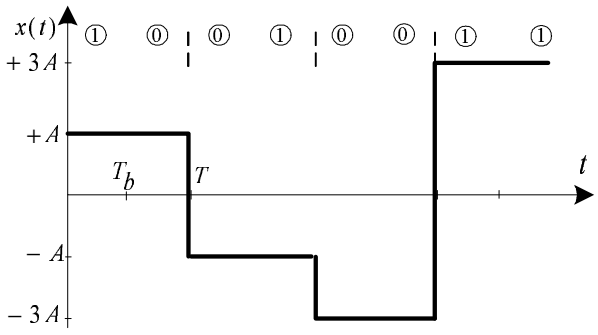


FIG. 5.14 – Signal correspondant au codage NRZ M-aire pour $M = 4$, $T = 2T_b$.

Si les bits sont indépendants et équiprobables, la moyenne de la séquence de symbole est nulle et la variance centrée $E((a_k)^2)$ est égale à :

$$\begin{aligned} E((a_k)^2) &= \frac{1}{M}\sum_{m=1}^{M}(2m-1-M)^2 \\ &= \frac{M^2-1}{3} \end{aligned} \tag{5.13}$$

Finalement, la formule de Bennett permet de calculer la densité spectrale de puissance du code NRZ M-aire :

$$\gamma_{XX}(f) = \frac{A^2 T}{3}(M^2-1)\left(\frac{\sin(\pi f T)}{\pi f T}\right)^2 \tag{5.14}$$

La densité spectrale de puissance du code NRZ M-aire pour $M = 4$ est comparée à celle du code NRZ sur la figure 5.15

5.3 Canal de Transmission

Le signal émis est de la forme :

$$x(t) = \sum_{k=-\infty}^{\infty} a_k g(t - kT) \tag{5.15}$$

Ce signal est ensuite modifié par le canal de transmission qui est en général modélisé par un filtre linéaire de réponse impulsionnelle $c(t)$. De plus, le canal ajoute un bruit blanc gaussien $b(t)$.

Le signal reçu $r(t)$ est égal à :

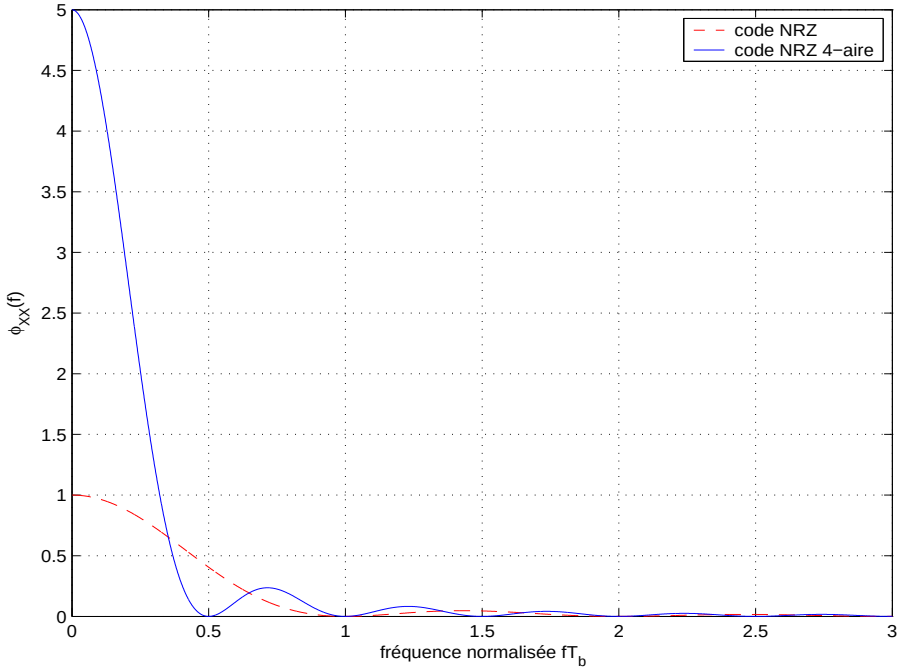
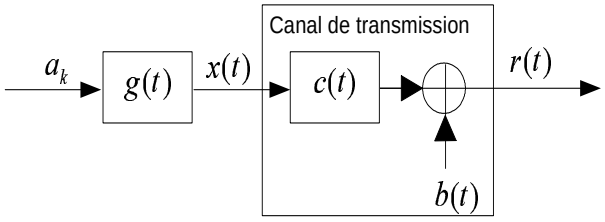


FIG. 5.15 – Densité spectrale des codes NRZ et NRZ-4 aire.

$$\begin{aligned} r(t) &= x(t) * c(t) + b(t) \\ &= \int_{-\infty}^{+\infty} c(\tau)x(t-\tau)d\tau + b(t) \end{aligned}$$



Nous verrons dans le prochain paragraphe que le signal reçu $r(t)$ est ensuite filtré par un filtre de réception de réponse impulsionnelle $h(t)$. Le signal $y(t)$ après filtrage s'écrit alors :

$$\begin{aligned} y(t) &= r(t) * h(t) \\ &= x(t) * c(t) * h(t) + n(t) \\ &= \sum_{k=-\infty}^{\infty} a_k g(t-kT) * c(t) * h(t) + n(t) \\ &= \sum_{k=-\infty}^{\infty} a_k p(t-kT) + n(t) \end{aligned} \tag{5.16}$$

où $*$ est le produit de convolution. $p(t) = g(t) * c(t) * h(t)$ est la réponse impulsionnelle de l'ensemble filtre de mise en forme+ canal de transmission + filtre de réception.

Dans le domaine fréquentiel, on a la relation $P(f) = G(f) \times C(f) \times H(f)$

5.3.1 Canal à bruit blanc additif gaussien

Le canal à bruit blanc additif gaussien (BBAG) est un canal de transmission dont la réponse en fréquence $C(f)$ est égale à 1 sur toute la bande du signal émis. Le canal est réduit à un simple additionneur de bruit blanc gaussien comme présenté sur la figure 5.16. Il permet de modéliser les canaux dont le bruit prédominant est un bruit thermique (canal de transmission et étage d'entrée du récepteur).

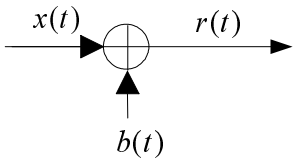


FIG. 5.16 – Modèle pour le signal reçu après un canal BBAG

On a la relation suivante entre l'entrée et la sortie du canal BBAG :

$$r(t) = x(t) + b(t) \tag{5.17}$$

$b(t)$ est un bruit blanc gaussien de densité spectrale de puissance unilatérale N_0. Sa fonction de covariance est donnée par :

$$\begin{aligned} R_{bb}(\tau) &= E\{b(t)b(t-\tau)\} \\ &= \frac{N_0}{2}\delta(\tau) \end{aligned} \tag{5.18}$$

où $\delta(.)$ représente l'impulsion de Dirac.

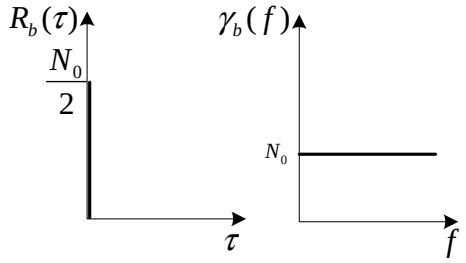


FIG. 5.17 – Fonction de covariance et densité spectrale de puissance du bruit blanc $b(t)$

Dans une bande de fréquence limitée B, $b(t)$ est modélisé par un processus aléatoire gaussien centré dont la densité de probabilité est la suivante :

$$p(b) = \frac{1}{\sqrt{2\pi N}} \exp\left(-\frac{b^2}{2N}\right) \tag{5.19}$$

La puissance de bruit N dans cette bande B est égale à $N_0 B$

5.4 Réception optimale pour le canal BBAG

5.4.1 Introduction

Dans ce paragraphe nous allons étudier la structure d'un récepteur optimal. On appelle récepteur l'ensemble composé des éléments démodulateur et détecteur. Dans ce chapitre, nous nous restreindrons au cas du détecteur délivrant des symboles estimés.

Nous concentrerons notre étude sur le récepteur cohérent c'est-à-dire que nous ferons l'hypothèse que les paramètres comme la fréquence et la phase des signaux reçus sont connus ou ont été correctement estimés.

5.4.2 Structure du modulateur

On suppose que le codeur de canal délivre des bits groupés par bloc de g bits. On a donc $M = 2^g$ messages différents possibles $c \in \{c_1, c_2, \ldots, c_M\}$.

Le rôle du modulateur consiste à associer à chaque message $c = c_i$ un signal $x_i(t)$, défini sur l'intervalle fini $0 \leq t \leq T$ et choisi parmi un jeu de $M = 2^g$ signaux.

Le schéma détaillé du modulateur est présenté sur la figure 5.18.

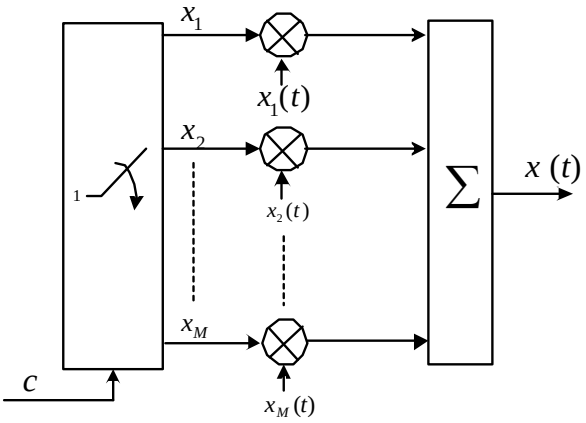


FIG. 5.18 – Schéma détaillé du modulateur.

Pour transmettre le signal $x_i(t)$ associé au message $c = c_i$, il suffit d'avoir $x_i = 1$ et $x_j = 0 \quad \forall j \neq i$

exemple 1 : cas du code en ligne non retour à zéro (NRZ) composé de $M = 2$ signaux élémentaires.

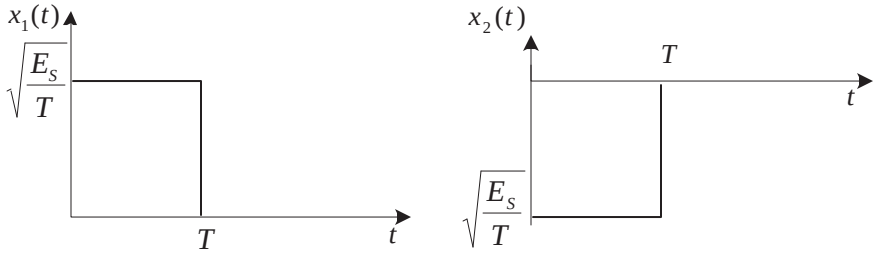


FIG. 5.19 – jeu des signaux $x_1(t)$ et $x_2(t)$ correspondant à l'exemple 1.

$$x(t) = \begin{cases} x_1(t) & \text{si} \quad c = 0 \quad (x_1 = 1, x_2 = 0) \\ x_2(t) & \text{si} \quad c = 1 \quad (x_1 = 0, x_2 = 1) \end{cases}$$

L'énergie de chaque signal $x_i(t)$ entre 0 et T est égale à $\mathcal{E}_{si}$

$$\mathcal{E}_{si} = \int_0^T x_i(t)^2 dt \tag{5.20}$$

exemple 2 : cas du jeu de $M = 4$ signaux biorthogonaux

$$x(t) = \begin{cases} x_1(t) & \text{si} \quad c = 00 \quad (x_1 = 1, x_2 = 0, x_3 = 0, x_4 = 0) \\ x_2(t) & \text{si} \quad c = 01 \quad (x_1 = 0, x_2 = 1, x_3 = 0, x_4 = 0) \\ x_3(t) & \text{si} \quad c = 10 \quad (x_1 = 0, x_2 = 0, x_3 = 1, x_4 = 0) \\ x_4(t) & \text{si} \quad c = 11 \quad (x_1 = 0, x_2 = 0, x_3 = 0, x_4 = 1) \end{cases}$$

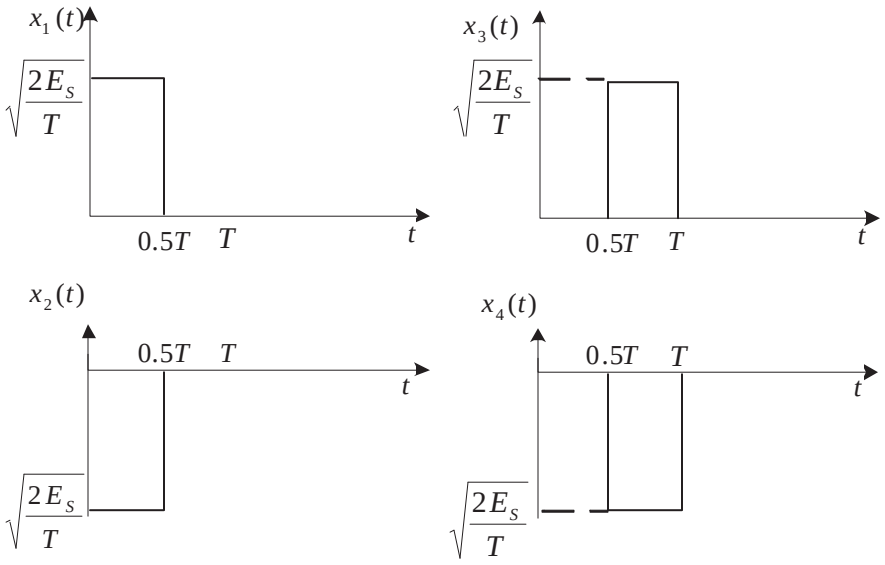


FIG. 5.20 – Jeu des signaux $x_i(t)$ $i = 1, 2, 3$ et 4 correspondant à l'exemple 2.

Une approche permettant de simplifier le modulateur consiste à représenter les M signaux possibles $x_i(t)$ par des combinaisons linéaires de N fonctions de base orthonormées $f_j(t)$ et d'énergie unitaire avec $N \leq M$. D'une manière générale, les fonctions $f_j(t)$ s'obtiennent en appliquant la procédure d'orthogonalisation de Gram-Schmidt.

Les signaux $x_i(t)$ peuvent s'exprimer par :

$$x_i(t) = \sum_{j=1}^{N} x_{ij} f_j(t) \tag{5.21}$$

où

$$x_{ij} = \int_0^T x_i(t) f_j(t) dt \tag{5.22}$$

et où les fonctions de base $f_1(t), f_2(t), \ldots, f_N(t)$ sont orthonormées :

$$\int_0^T f_j(t) f'_j(t) dt = \delta_{j'j} = \begin{cases} 1 & \text{si} \quad j = j' \\ 0 & \text{si} \quad j \neq j' \end{cases} \tag{5.23}$$

Ainsi l'énergie des fonctions de base entre 0 et T est égale à 1.

L'énergie de chaque signal $x_i(t)$ entre 0 et T est égale à $\mathcal{E}_{si}$

$$\mathcal{E}_{si} = \int_0^T x_i(t)^2 dt = \sum_{j=1}^{N} x_{ij}^2 \tag{5.24}$$

L'énergie moyenne $\mathcal{E}_s$ est donnée par :

$$\mathcal{E}_s = \frac{1}{M}\mathcal{E}_{si} \tag{5.25}$$

En utilisant cette représentation on obtient le schéma donné sur la figure 5.21.

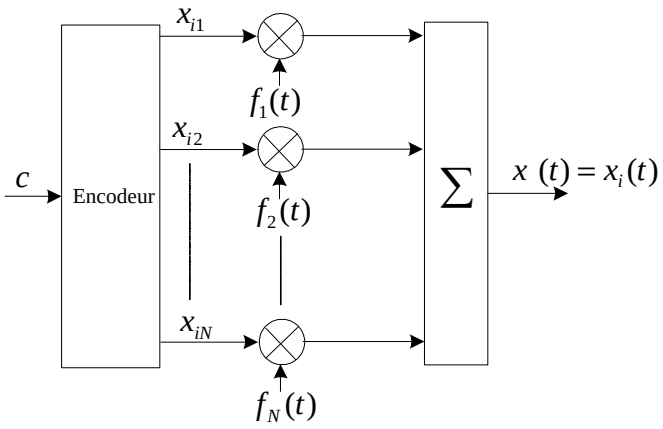


FIG. 5.21 – Schéma détaillé du démodulateur.

exemple 1(suite) : cas du code en ligne NRZ $M = 2$ signaux

Dans cet exemple, N=1 car une seule fonction $f_1(t)$ donnée sur la figure 5.22 suffit pour générer les deux signaux $x_1(t)$ et $x_2(t)$ $(N = 1)$.

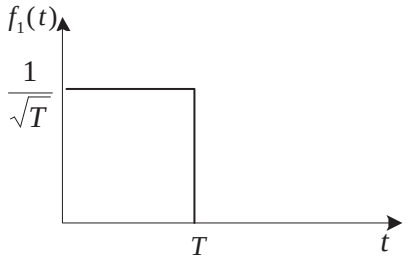


FIG. 5.22 – signal $f_1(t)$ pour l'exemple 1.

$$x(t) = \begin{cases} x_1(t) = -\sqrt{\mathcal{E}_s} f_1(t) & \text{si} \quad c = 0 \quad (x_{11} = -\sqrt{\mathcal{E}_s}) \\ x_2(t) = +\sqrt{\mathcal{E}_s} f_1(t) & \text{si} \quad c = 1 \quad (x_{21} = +\sqrt{\mathcal{E}_s}) \end{cases}$$

La structure du modulateur est donnée sur la figure 5.23.

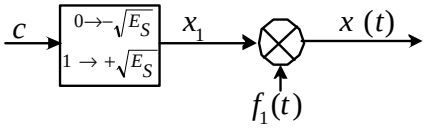


FIG. 5.23 – Schéma du modulateur NRZ

Dans cet exemple, la fonction de base $f_1(t)$ correspond à la réponse impusionnelle $g(t)$ du filtre d'émission. Le schéma du modulateur correspondant est présenté sur la figure 5.24

exemple 2 (suite) : cas du jeu de $M = 4$ signaux biorthogonaux

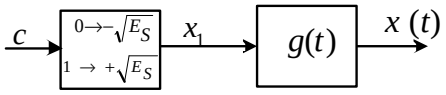


FIG. 5.24 – Schéma du modulateur NRZ

Ici, N=2 est les deux fonctions $f_1(t)$ et $f_2(t)$ présentées sur la figure 5.25 permettent de générer les 4 signaux ($N = 2$).

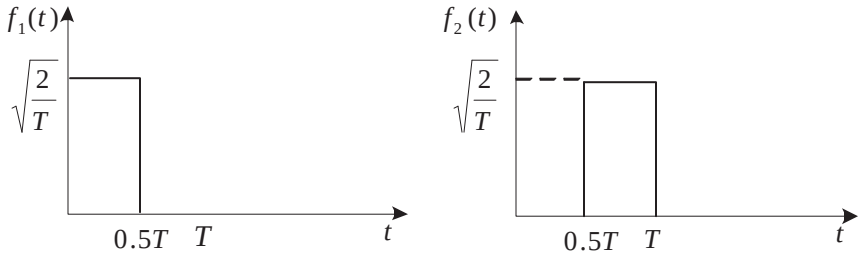


FIG. 5.25 – jeu des signaux $f_1(t)$ et $f_2(t)$ pour l'exemple 2.

$$x(t) = \begin{cases} x_1(t) = -\sqrt{\mathcal{E}_s} f_1(t) & \text{si} \quad c = 00 \quad (x_{11} = -\sqrt{\mathcal{E}_s}, x_{12} = 0) \\ x_2(t) = +\sqrt{\mathcal{E}_s} f_1(t) & \text{si} \quad c = 01 \quad (x_{21} = +\sqrt{\mathcal{E}_s}, x_{22} = 0) \\ x_3(t) = -\sqrt{\mathcal{E}_s} f_2(t) & \text{si} \quad c = 10 \quad (x_{31} = 0, x_{32} = -\sqrt{\mathcal{E}_s}) \\ x_4(t) = +\sqrt{\mathcal{E}_s} f_2(t) & \text{si} \quad c = 11 \quad (x_{41} = 0, x_{42} = +\sqrt{\mathcal{E}_s}) \end{cases}$$

5.4.3 Récepteur optimal pour un canal à bruit blanc additif gaussien

On considère qu'un des M signaux possibles $x_i(t)$ a été transmis sur un canal à bruit blanc additif gaussien (BBAG) pendant la durée T.

En entrée du récepteur, on a

$$r(t) = x_i(t) + b(t) \qquad 0 \leq t \leq T \tag{5.26}$$

où $b(t)$ est un bruit blanc gaussien de densité spectrale de puissance unilatérale N_0

La réception optimale comprend deux phases : la démodulation optimale qui détermine la séquence maximisant le rapport signal à bruit et la détection qui à partir de cette séquence estime c.

Il existe deux structures équivalentes de démodulateur optimal : le corrélateur et le filtre adapté

Corrélateur

Le principe du corrélateur consiste à projeter le signal reçu sur les N fonctions de base $f_j(t)$.

Le schéma de ce nouveau démodulateur est donné sur la figure 5.26.

Comme les fonctions de base $f_j(t)$ sont d'énergie unitaire entre 0 et T on a relation suivante :

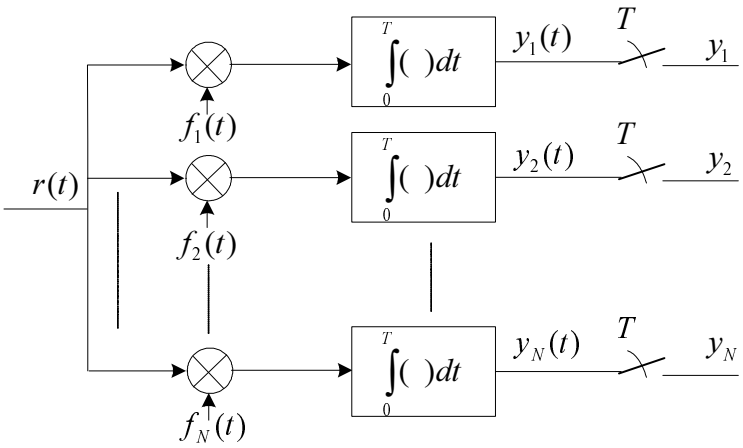


FIG. 5.26 – Schéma du démodulateur utilisant les fonctions de base

$$\begin{aligned} y_j &= y_j(T) \\ &= \int_0^T r(t) f_j(t) dt \\ &= \int_0^T x_i(t) f_j(t) dt + \int_0^T b(t) f_j(t) dt \\ &= \int_0^T \sum_{j'=1}^{N} x_{ij'} f_{j'}(t) f_j(t) dt + \int_0^T b(t) f_j(t) dt \\ &= \sum_{j'=1}^{N} x_{ij'} \int_0^T f_{j'}(t) f_j(t) dt + \int_0^T b(t) f_j(t) dt \\ &= x_{ij} + n_j \end{aligned} \tag{5.27}$$

avec y_j la sortie du i^{eme} échantillonneur à l'instant T et $1 \leq j \leq N$.

Le signal reçu $r(t)$ est maintenant représenté par un vecteur à N composantes y_j. Les termes n_j sont des variables aléatoires relatives au bruit additif du canal de transmission. Ces variables aléatoires sont gaussiennes car le filtrage linéaire d'un bruit gaussien filtre linéaire ne modifie pas son caractère gaussien. Calculons sa moyenne :

$$E(n_j) = \int_0^T b(t) f_j(t) dt = 0 \tag{5.28}$$

et sa variance

$$\begin{aligned}\sigma_n^2 &= E\big[n_j^2\big] \\ &= E\Big[\int_0^T \int_0^T b(t)b(u)f(t)f(u)dtdu\Big] \\ &= \frac{N_0}{2}\int_0^T \int_0^T \delta(t-u)f(t)f(u)dtdu \\ &= \frac{N_0}{2}\int_0^T f(t)^2 dt \\ &= \frac{N_0}{2}\end{aligned} \tag{5.29}$$

Ainsi, les échantillons de bruit sont centrés et de variance $\sigma^2 = \frac{N_0}{2}$. Ces échantillons sont également décorrélés et indépendants.

Finalement il nous restera à décider en faveur du signal le plus probablement émis.

On dit que les échantillons $y_1, \ldots, y_N$ forment **un résumé exhaustif** du signal $r(t)$. Ainsi, on peut résumer l'ensemble de la chaîne de transmission par les N équations (5.27). On utilisera également cette propriété pour simuler les chaînes de communications numériques

exemple 1(suite) : pour le signal NRZ, nous avons vu qu'une seule fonction de base $f_1(t)$ est nécessaire. A partir du schéma de la figure 5.26, on dérive le schéma présenté sur la figure 5.27.

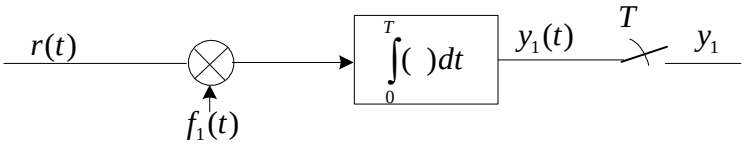


FIG. 5.27 – Schéma du démodulateur NRZ

Soit y_1 la sortie de l'échantillonneur à l'instant T. Comme précédemment, nous avons la relation :

$$\begin{aligned}y_1 &= y_1(T) \\ &= \int_0^T r(t)f_1(t)dt \\ &= \int_0^T x_i(t)f_1^2(t)dt + \int_0^T b(t)f_1(t)dt \\ &= x_{i1} + n\end{aligned} \tag{5.30}$$

Le bruit n en sortie de l'échantillonneur est gaussien de moyenne nulle $E(n) = 0$ et de variance $\sigma^2 = \frac{N_0}{2}$.

Pour le signal NRZ, le rapport signal à bruit SNR après échantillonnage est :

$$SNR = \frac{P_x}{P_n} = \frac{E\big[x_i^2\big]}{\sigma_n^2} = \frac{\sum_i Pr(X = x_i)x_i^2}{\sigma_n^2} = \frac{x_i^2}{\sigma_n^2} \tag{5.31}$$

Comme $x_i^2 = \mathcal{E}_s$, et $\sigma_n^2 = \frac{N_0}{2}$, on obtient finalement

$$SNR = \frac{2\mathcal{E}_s}{N_0} \tag{5.32}$$

Sur la figure 5.28 nous présentons la forme des signaux en sortie du modulateur, du canal et du corrélateur.

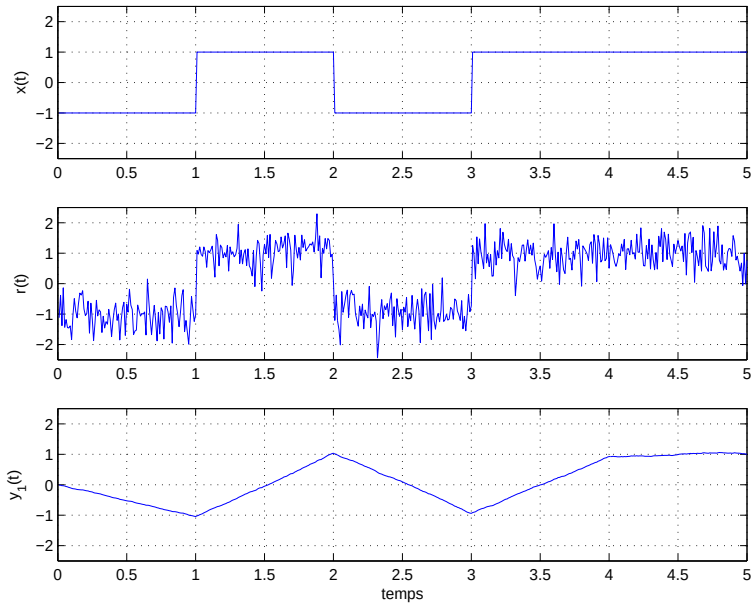


FIG. 5.28 – Signaux en sortie du modulateur, du canal et du corrélateur

Filtre adapté

Au lieu d'utiliser des corrélateurs à la réception pour obtenir les échantillons y_j, il est possible d'utiliser des filtres dont la réponse impulsionnelle est $h_j(t)$. La relation entre $h_j(t)$ et les fonctions de base $f_j(t)$ est la suivante :

$$h_j(t) = f_j(T-t) \qquad \text{avec} \quad 0 \leq t \leq T \tag{5.33}$$

La sortie de chaque filtre s'exprime comme suit :

$$\begin{aligned} y_j(t) &= r(t) * h_j(t) \\ &= \int_{-\infty}^{+\infty} r(t) h_j(t-\tau) d\tau \\ &= \int_{0}^{t} r(t) h_j(t-\tau) d\tau \\ &= \int_{0}^{t} r(t) f_j(T-t+\tau) d\tau \end{aligned}$$

Si on échantillonne la sortie des filtres à l'instant T, on retrouve la relation précédente :

$$\begin{aligned} y_j(T) &= \int_{0}^{T} r(t) f_j(\tau) d\tau \\ &= y_j \end{aligned} \tag{5.34}$$

Nous avons ainsi montré que la sortie des filtres $h_j(t)$ à l'instant $t = T$ est identique à celle de la structure avec corrélateur.

Les filtres de réponse impulsionnelle $h_j(t)$ sont appelés filtres adaptés aux fonctions de base $f_j(t)$ définies pour $0 \leq t \leq T$.

On peut démontrer que l'utilisation de filtres adaptés permet de maximiser le rapport signal à bruit et par conséquence de minimiser le taux d'erreurs.

Le schéma du démodulateur est donné sur la figure 5.29.

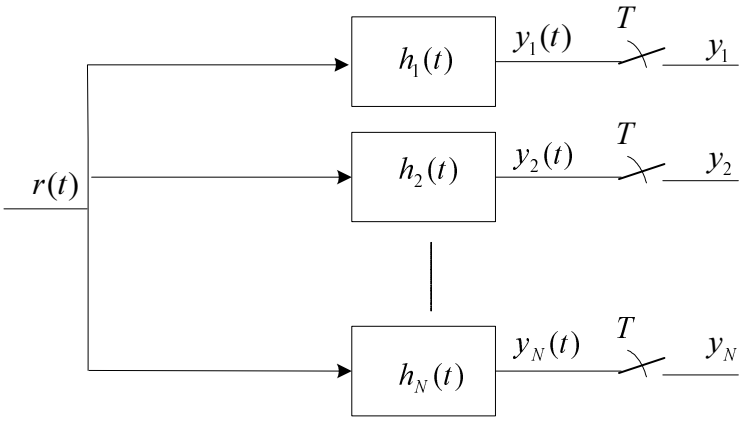


FIG. 5.29 – Schéma du démodulateur avec filtres adaptés

Il est important de souligner que le rapport signal à bruit en sortie d'un filtre adapté ne dépend pas de la forme du signal mais de son énergie !

exemple 1(suite) :

Pour le cas du code en ligne NRZ, la réponse impulsionnelle $h(t)$ du filtre adapté est la suivante :

$$h(t) = g(T - t) \tag{5.35}$$

où $g(t)$ est la réponse impulsionnelle du filtre d'émission.

Le schéma de la chaîne avec filtre adapté est présenté sur la figure 5.30

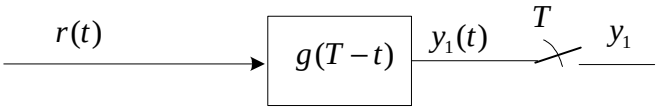


FIG. 5.30 – Schéma de la chaîne NRZ avec filtre adapté

L'utilisation d'un filtre adapté à la place du corrélateur donne exactement le même résultat.

Détecteur optimal

L'objectif du détecteur optimal est de déterminer le symbole qui a été le plus vraisemblablement émis $\hat{x}$.

Soit le symbole x envoyé dans un canal discret stationnaire sans mémoire de densité de probabilité conditionnelle $p(y/x)$ et y l'échantillon reçu après filtrage adapté.

Un détecteur *maximum a posteriori* (MAP) cherche parmi tous les symboles possibles x, le symbole estimé $\hat{x}$ pour lequel la probabilité conditionnelle $Pr(x|y)$ est la plus grande.

$$\hat{x} = \arg\max_x Pr(x|y) \tag{5.36}$$

En utilisant la loi de Bayes, on peut écrire :

$$Pr(x|y) = \frac{Pr(y|x)Pr(x)}{Pr(y)} \tag{5.37}$$

Si nous faisons l'hypothèse que tous les symboles sont équiprobables, et comme le dénominateur $Pr(y)$ est commun à tous les symboles, le message estimé $\hat{x}$ est le message pour lequel la probabilité conditionnelle $Pr(y|x)$ est la plus grande.

$$\hat{x} = \arg\max_x Pr(y|x) \tag{5.38}$$

Un détecteur utilisant ce critère est appelé un détecteur à *maximum de vraisemblance* (*maximum likelihood* en anglais ou ML).

La recherche du message le plus probable implique donc que le détecteur ML calcule les distances euclidiennes entre l'échantillon reçu et les échantillons correspondant à tous les symboles possibles.

Ainsi, lorsque les messages sont équiprobables, les détecteurs MAP et ML sont identiques.

5.4.4 Calcul du taux d'erreurs binaires pour un signal NRZ sur un canal à bruit blanc additif gaussien

Pour évaluer les performances d'une chaîne de transmission numérique, il est important de déterminer le taux d'erreurs binaires en fonction du rapport signal à bruit. Dans ce paragraphe, nous considérerons le cas d'un code en ligne NRZ.

Soit une transmission par codage en ligne NRZ sur un canal BBAG, nous avons vu dans le paragraphe précédent qu'il est possible d'exprimer la sortie échantillonnée y du récepteur optimal comme suit :

$$y = x + n \tag{5.39}$$

où $x = \pm\sqrt{E_b}$ car $T = T_b$. On rappelle que dans le cas du code NRZ, l'énergie par bit est égale à l'énergie par symbole $E_b = E_s$.

Comme le bruit est stationnaire, n est une variable aléatoire gaussienne d'écart type σ et centrée dont la densité de probabilité est la suivante :

$$p(n) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{ - \frac{n^2}{2\sigma^2} \right\} \tag{5.40}$$

Le détecteur ML réalise simplement l'opération de décision suivante :

$$\hat{c} = \begin{cases} 0 & \text{si} \quad y \leq s \\ 1 & \text{si} \quad y > s \end{cases} \tag{5.41}$$

Cette opération se résume ici à un simple seuillage par rapport au niveau s.

La probabilité que l'amplitude de l'échantillon reçu y soit inférieure à s sachant que $c = 1$ (et donc $\hat{x} = +\sqrt{E_b}$ est égale à :

$$p(y < s|x = +\sqrt{E_b}) = \frac{1}{\sqrt{2\pi\sigma^2}} \int_{-\infty}^{s} \exp\left\{ - \frac{(y - \sqrt{E_b})^2}{2\sigma^2} \right\} dy \tag{5.42}$$

Cette probabilité est égale à la probabilité de décider en faveur de $\hat{x} = -\sqrt{E_b}$ alors que $x = +\sqrt{E_b}$ a été transmis. Cette probabilité d'erreurs par paire est notée $p(x = +\sqrt{E_b} \rightarrow \hat{x} = -\sqrt{E_b})$

$$p(x = +\sqrt{E_b} \rightarrow \hat{x} = -\sqrt{E_b}) = p(y < s|x = +\sqrt{E_b}) \tag{5.43}$$

De même, la probabilité que l'amplitude de l'échantillon y soit supérieure au seuil de décision s sachant que $c = 0$ (et donc $x = -\sqrt{E_b}$) est égale à :

$$p(y > s|x = -\sqrt{E_b}) = \frac{1}{\sqrt{2\pi\sigma^2}} \int_s^{+\infty} \exp\left\{ - \frac{(y + \sqrt{E_b})^2}{2\sigma^2} \right\} dy \tag{5.44}$$

La probabilité d'erreurs par bit ou taux d'erreurs bit (TEB) est alors le suivant :

$$\begin{aligned} TEB &= Pr(\text{erreur bit}) \\ &= \sum_x Pr(x = a, \text{erreur bit}) \\ &= \sum_x Pr(x = a) Pr(\text{erreur bit} \mid \text{x=a}) \\ &= Pr(x = +\sqrt{E_b}) p(x = +\sqrt{E_b} \to \hat{x} = -\sqrt{E_b}) + Pr(x = -\sqrt{E_b}) p(x = -\sqrt{E_b} \to \hat{x} = +\sqrt{E_b}) \end{aligned} \tag{5.45}$$

Nous ferons l'hypothèse que les bits émis sont équiprobables. On a :

$$Pr(c = 1) = Pr(c = 0) = Pr(x = +\sqrt{E_b}) = Pr(x = -\sqrt{E_b}) = 0.5 \tag{5.46}$$

La densité de probabilité $p(y)$ a alors la forme suivante :

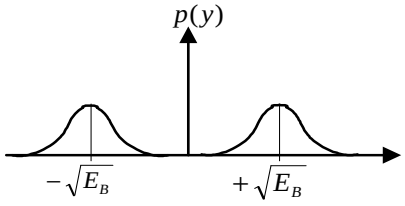


FIG. 5.31 – densité de probabilité $p(y)$

Le seuil de décision est donc placé exactement entre $+\sqrt{E_b}$ et $-\sqrt{E_b}$: $s = 0$.
Le taux d'erreurs binaires devient alors :

$$\begin{aligned} TEB &= 0.5p(x = +\sqrt{E_b} \to \hat{x} = -\sqrt{E_b}) + 0.5p(x = -\sqrt{E_b} \to \hat{x} = +\sqrt{E_b}) \\ &= p(x = -\sqrt{E_b} \to \hat{x} = +\sqrt{E_b}) \\ &= \frac{1}{\sqrt{2\pi\sigma^2}} \int_0^{+\infty} \exp\left\{ - \frac{(y + \sqrt{E_b})^2}{2\sigma^2} \right\} dy \end{aligned} \tag{5.47}$$

Faisons un changement de variable $z = \frac{y+\sqrt{E_b}}{\sqrt{2}\sigma}$, $dz = \frac{dy}{\sqrt{2}\sigma}$. On obtient alors :

$$TEB = \frac{1}{\sqrt{\pi}} \int_{\sqrt{E_b/2\sigma^2}}^{+\infty} \exp\left\{ - z^2 \right\} dz \tag{5.48}$$

Fonction d'erreurs
On trouve dans la littérature deux fonctions d'erreurs :
• l'aire soutendue par la courbe normale canonique de a à ∞ :

$$Q(a) = \frac{1}{\sqrt{2\pi}} \int_a^{\infty} \exp\left\{ - \frac{z}{2} \right\} dz \tag{5.49}$$

• les fonctions erf et erfc (erf complémentaire) :

$$\text{erf}(a) = \frac{2}{\sqrt{\pi}} \int_{-\infty}^{a} \exp(-z^2)dz \tag{5.50}$$

$$\text{erfc}(a) = 1 - \text{erf}(a) = \frac{2}{\sqrt{\pi}} \int_{a}^{+\infty} \exp(-z^2)dz \tag{5.51}$$

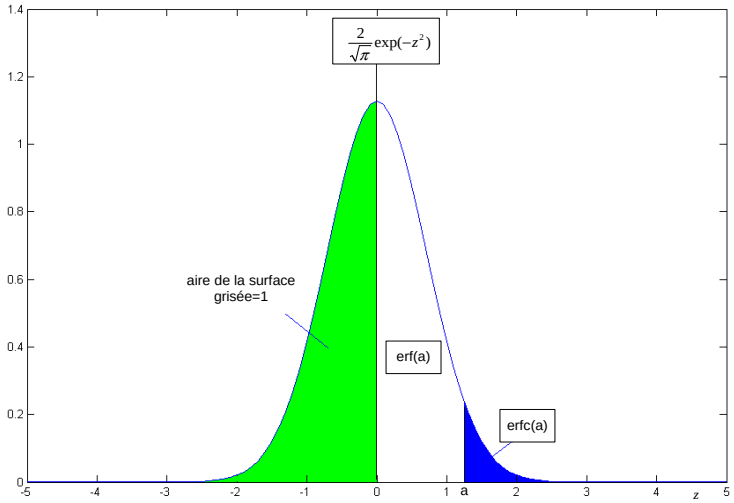


FIG. 5.32 – fonctions erf et erfc

La fonction erfc(a) existe dans Matlab sous le nom : **erfc**(a).
On passe de la fonction erfc(a) à la fonction $Q(a)$ en utilisant la relation suivante :

$$Q(a) = \frac{1}{2}\text{erfc}\left(\frac{a}{\sqrt{2}}\right) \tag{5.52}$$

En utilisant la relation (5.48) et $\sigma^2 = \frac{N_0}{2}$, on obtient finalement la relation suivante entre le taux d'erreurs bit et le rapport signal à bruit :

$$TEB = \frac{1}{2}\text{erfc}\left\{\sqrt{\frac{\mathcal{E}_b}{N_0}}\right\} \tag{5.53}$$

La courbe du taux d'erreurs bit en fonction du rapport E_b/N_0 est présentée sur la figure (5.33).

5.4.5 Probabilité d'erreurs par paire : cas scalaire

Soient deux symboles x_i et x_j dont la distance euclidienne est $d(x_i, x_j)$. Pour un canal BBAG, la probabilité $Pr(x_i \rightarrow x_j)$ que y soit plus près de x_j que du symbole transmis x_i est donnée par :

$$Pr(x_i \rightarrow x_j) = \frac{1}{2}\text{erfc}\left(\frac{d(x_i, x_j)}{2\sqrt{N_0}}\right) \tag{5.54}$$

Dans le cas du code NRZ, la distance euclidienne est égale à $2\sqrt{E_b}$. On retrouve bien l'expression du taux d'erreurs binaires décrit par l'équation (5.53) lorsque les symboles sont équiprobables.

a	erfc(a)	a	erfc(a)	a	erfc(a)
0.0	1.0000e+000	1.7	1.6210e-002	3.4	1.5220e-006
0.1	8.8754e-001	1.8	1.0909e-002	3.5	7.4310e-007
0.2	7.7730e-001	1.9	7.2096e-003	3.6	3.5586e-007
0.3	6.7137e-001	2.0	4.6777e-003	3.7	1.6715e-007
0.4	5.7161e-001	2.1	2.9795e-003	3.8	7.7004e-008
0.5	4.7950e-001	2.2	1.8628e-003	3.9	3.4792e-008
0.6	3.9614e-001	2.3	1.1432e-003	4.0	1.5417e-008
0.7	3.2220e-001	2.4	6.8851e-004	4.1	6.7000e-009
0.8	2.5790e-001	2.5	4.0695e-004	4.2	2.8555e-009
0.9	2.0309e-001	2.6	2.3603e-004	4.3	1.1935e-009
1.0	1.5730e-001	2.7	1.3433e-004	4.4	4.8917e-010
1.1	1.1979e-001	2.8	7.5013e-005	4.5	1.9662e-010
1.2	8.9686e-002	2.9	4.1098e-005	4.6	7.7496e-011
1.3	6.5992e-002	3.0	2.2090e-005	4.7	2.9953e-011
1.4	4.7715e-002	3.1	1.1649e-005	4.8	1.1352e-011
1.5	3.3895e-002	3.2	6.0258e-006	4.9	4.2189e-012
1.6	2.3652e-002	3.3	3.0577e-006	5.0	1.5375e-012

TAB. 5.1 – table de la fonction erfc.

5.4.6 Calcul du taux d'erreurs symboles pour un signal NRZ multi-niveaux

On considère que les symboles prennent leurs valeurs dans l'alphabet $\{\pm A, \pm 3A, \ldots, \pm(M-1)A\}$. L'amplitude des symboles est alors définie comme suit :

$$A_m = (2m - 1 - M)A \qquad \text{avec} \qquad m = 1, 2, \ldots, M \tag{5.55}$$

Exemple : $M = 4$

L'énergie moyenne par symbole est égale à :

$$\begin{aligned} E_s &= \frac{1}{M}\sum_{m=1}^{M}(A_m)^2 \\ &= \frac{A^2}{M}\sum_{m=1}^{M}(2m-1-M)^2 \\ &= A^2\frac{M^2-1}{3} \end{aligned} \tag{5.56}$$

Le calcul exact du taux d'erreurs symbole est assez compliqué car il est nécessaire de tenir compte de tous les types d'erreurs possibles. Nous nous contenterons d'une approximation ne tenant compte que des cas où l'erreur symbole provient d'une décision en faveur du symbole adjacent. On a $2(M-1)$ paires de points adjacents et la distance euclidienne entre ces points est égale à $2A$. Ainsi, le taux d'erreurs symbole (TES) est égal à :

$$\begin{aligned} TES &= \frac{2(M-1)}{2M} erfc\left(\frac{2A}{2\sqrt{N_0}}\right) \\ &= \frac{M-1}{M} erfc\left(\sqrt{\frac{3}{M^2-1}\frac{E_S}{N_0}}\right) \\ &= \frac{M-1}{M} erfc\left(\sqrt{\frac{3log_2 M}{M^2-1}\frac{E_b}{N_0}}\right) \end{aligned} \tag{5.57}$$

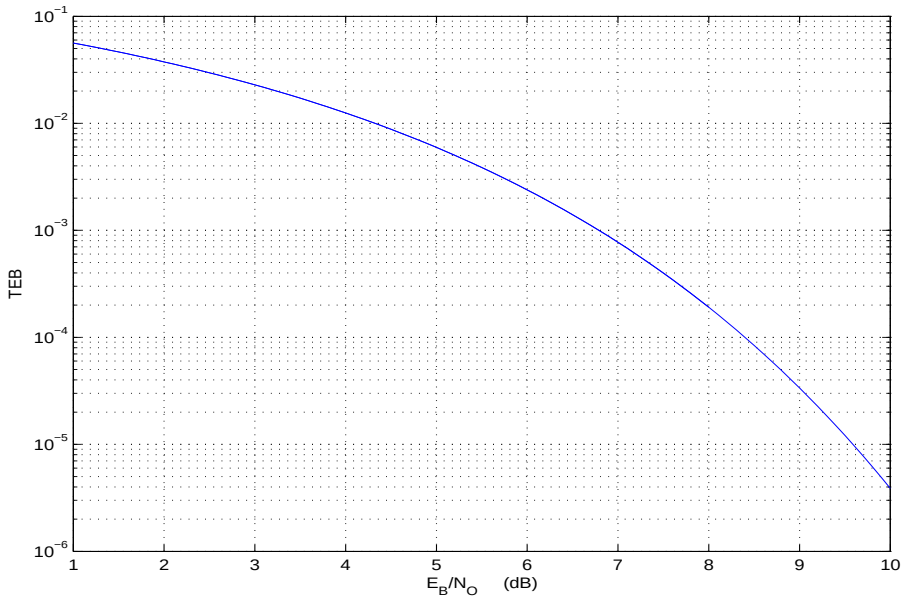


FIG. 5.33 – $TEB = f(E_b/N_0)$ pour un code NRZ sur canal BBAG.

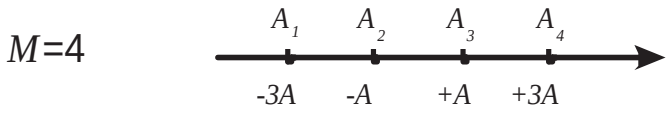


FIG. 5.34 – Amplitude des symboles pour un signal NRZ $M = 4$ niveaux.

Ce résultat est à comparer à celui du code NRZ.

Pour déduire le taux d'erreurs bit à partir du taux d'erreurs symbole, il faut déterminer le nombre d'erreurs bit qu'engendre une erreur symbole. Nous verrons que lorsqu'un codage de Gray est appliqué sur les symboles, une erreur symbole n'implique qu'une erreur bit et on a alors :

$$TEB = \frac{TES}{log_2 M} \tag{5.58}$$

5.5 Critère de Nyquist

5.5.1 Introduction

Jusqu'à présent nous avons considéré des transmissions numériques sur des canaux à bande passante infinie. Dans beaucoup d'applications (modems téléphoniques, liaisons hertziennes et satellitaires, communications radiomobiles, ...), la largeur de bande est limitée et le problème consiste à transmettre le débit le plus élevé possible dans une bande de fréquence donnée.

Si la bande du signal émis est limitée, la forme de l'impulsion élémentaire $g(t)$ est de durée infinie. Le problème consiste donc à choisir cette forme d'onde pour pouvoir reconstituer parfaitement à la réception les échantillons émis à la cadence symbole de $1/T$.

5.5.2 Interférence entre symboles

Nous avons vu que la sortie du filtre de réception peut s'écrire

$$\begin{aligned} y(t) &= r(t) * h(t) \\ &= x(t) * c(t) * h(t) + n(t) \\ &= \sum_{i=-\infty}^{\infty} a_i g(t - iT) * c(t) * h(t) + n(t) \\ &= \sum_{i=-\infty}^{\infty} a_i p(t - iT) + n(t) \end{aligned} \tag{5.59}$$

avec $p(t) = g(t) * c(t) * h(t)$ et $n(t) = b(t) * h(t)$

On échantillonne $y(t)$ aux instants $t = kT + \tau$ (τ est un délai permettant d'ajuster l'instant optimal d'échantillonnage)

$$\begin{aligned} y(kT) &= \sum_{i=-\infty}^{+\infty} a_i p(kT - iT) + n(kT) \\ &= a_k p(0) + \sum_{i \neq k}^{+\infty} a_i p((k-i)T) + n(kT) \end{aligned} \tag{5.60}$$

Cette expression est composée de 3 termes :

Le premier terme est proportionnel au k^{ieme} symbole transmis

Le second terme est la contribution de tous les autres symboles transmis sur l'échantillon $y(kT)$. Ce terme est appelé l'interférence intersymbole.

Le troisième terme est la contribution du bruit

Comme le second et le troisième terme diminuent les performances du système de transmission, nous devrons choisir les filtres d'émission et de réception afin de les minimiser.

5.5.3 Diagramme de l'œil

Le diagramme de l'œil permet de visualiser les interférences intersymboles. Il consiste à superposer toutes les sections de durée T du signal $y(t)$. Un exemple est présenté sur la figure 5.35 pour un signal NRZ filtré par un filtre en cosinus surélevé.

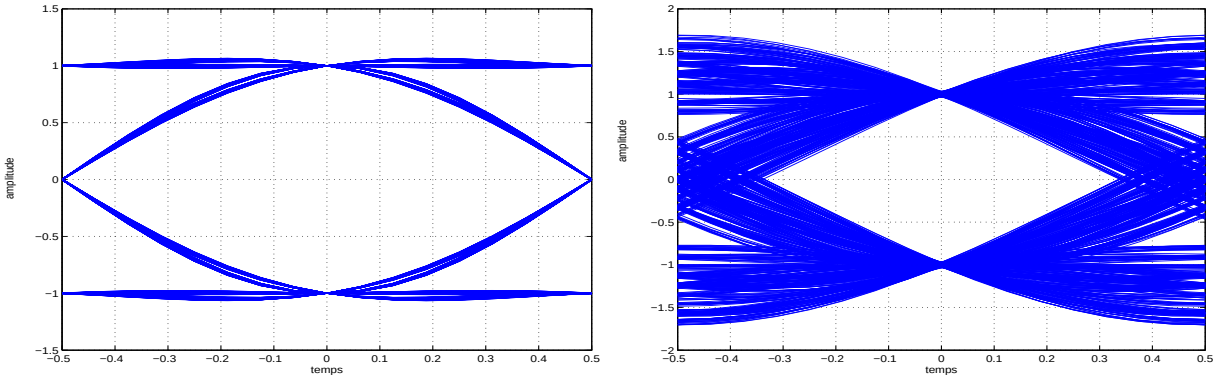


FIG. 5.35 – Diagramme de l'œil d'un signal NRZ filtré par un filtre en cosinus surélevé $\alpha = 1$ et $\alpha = 0,35$

La figure 5.36 illustre graphiquement l'effet de l'interférence intersymbole et du bruit sur le diagramme de l'œil. On peut noter en particulier que même en absence de bruit, l'interférence intersymbole tend à fermer le diagramme de l'œil et donc à rendre plus difficile la décision (amplitude et instant d'échantillonnage).

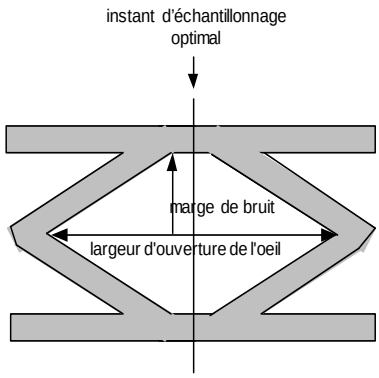


FIG. 5.36 – Diagramme de l'œil

5.5.4 Critère de Nyquist

Pour garantir l'absence d'interférence intersymbole, on doit avoir la condition suivante sur la forme d'onde $p(t)$:

$$p(kT) = \begin{cases} 1 & \text{pour} \quad k = 0 \\ 0 & \text{pour} \quad k \neq 0 \end{cases} \tag{5.61}$$

On peut vérifier qu'avec cette condition, l'équation (5.60) devient :

$$y(kT) = a_k + n(kT) \tag{5.62}$$

Dans le domaine fréquentiel, le critère de Nyquist devient [24] :

$$\sum_{i=-\infty}^{+\infty} P\left(f + \frac{i}{T}\right) = T \tag{5.63}$$

Soit B la bande passante du canal de transmission ($C(f) = 0$ pour f > B).

Discutons de la faisabilité de réaliser le critère selon la largeur de bande B par rapport à la durée de la période symbole T comme présenté sur la figure 5.37

- Si $\frac{1}{2T} > B$, alors il n'existe pas de filtre $G(f)$ et $H(f)$ tel que $P(f) = G(f)C(f)H(f)$ satisfait au critère de Nyquist

- Si $\frac{1}{2T} = B$, il existe une solution possible : $P(f)$ doit être la réponse d'un filtre passe-bas parfait de fréquence de coupure B :

$$P(f) = \begin{cases} T & \text{si} \quad |f| < B = \frac{1}{2T} \\ 0 & \text{sinon} \end{cases} \tag{5.64}$$

La réponse impulsionnelle associée est :

$$p(t) = \frac{\sin(\pi t/T)}{\pi t/T} \tag{5.65}$$

Elle est présentée sur la figure 5.38. Il faut préciser que le filtre passe bas parfait n'est pas physiquement réalisable.

- Si $\frac{1}{2T} < B$, alors il existe une famille de filtre $G(f)$ et $H(f)$ tel que $P(f) = G(f)C(f)H(f)$ répond au critère de Nyquist. Les filtres appartenant à cette famille doivent satisfaire la condition suivante :

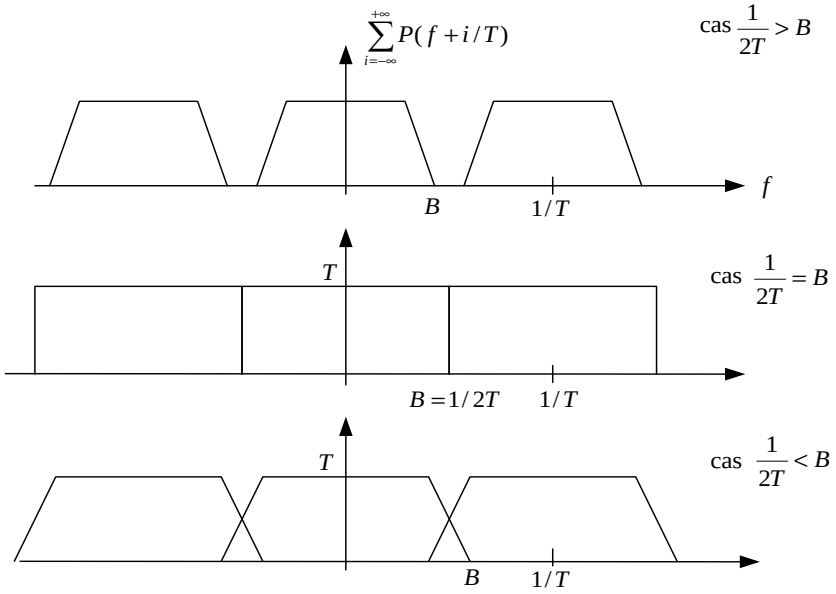


FIG. 5.37 – Réponses $\sum P(f + i/T)$ pour $1/2T > B$, $1/2T = B$ et $1/2T < B$

$$P(f) + P\left(f - \frac{1}{T}\right) = T \tag{5.66}$$

5.5.5 Le filtre en cosinus surélevé

La fonction de transfert d'un filtre en cosinus surélevé est la suivante

$$P(f) = \begin{cases} T & \text{si} \quad 0 \leq |f| \leq \frac{1-\alpha}{2T} \\ T\cos^2\left\{\frac{\pi}{4\alpha}(2fT - (1-\alpha))\right\} & \text{si} \quad \frac{1-\alpha}{2T} < |f| < \frac{1+\alpha}{2T} \\ 0 & \text{si} \quad |f| \geq \frac{1+\alpha}{2T} \end{cases} \tag{5.67}$$

α est le facteur d'arrondi (*roll-off* en anglais) et est compris entre 0 (filtre passe-bas parfait - diagramme de l'œil le plus fermé) et 1 (bande passante maximale - diagramme de l'œil le plus ouvert). En pratique cette valeur est choisie entre 0.22 et 0.35.

La réponse en fréquence du filtre en cosinus surélevé est présentée sur la figure 5.39.

On peut vérifier que cette fonction de transfert satisfait le critère (5.66).

La réponse impulsionnelle $p(t)$ est la suivante :

$$p(t) = \frac{\sin(\pi t/T)}{\pi t/T} \frac{\cos(\alpha\pi t/T)}{1 - 4\alpha^2 t^2/T^2} \tag{5.68}$$

Lorsque le canal de transmission est un canal à bruit blanc additif gaussien ($C(f) = 1$ dans bande considérée), on a $P(f) = G(f)H(f)$. Afin de maximiser le rapport signal à bruit à la réception, nous avons vu que le filtre de réception doit être le filtre adapté au filtre d'émission. Ainsi, en pratique, on scinde le filtre en cosinus surélevé en deux filtres identiques appelés filtres en racine de cosinus surélevé :

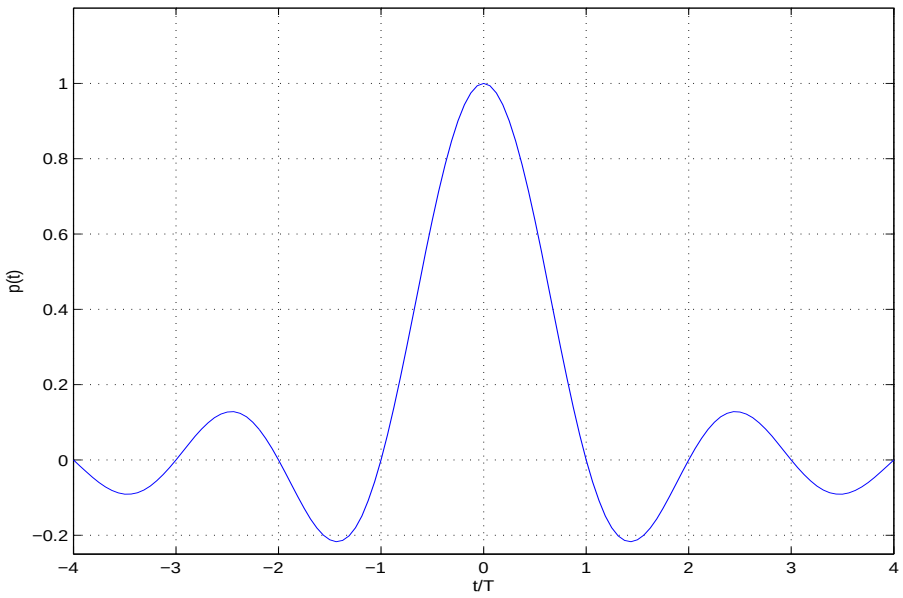


FIG. 5.38 – réponse impulsionnelle du filtre passe-bas parfait

$$G(f) = H(f) = \begin{cases} \sqrt{T} & \text{si} \quad 0 \leq |f| \leq \frac{1-\alpha}{2T} \\ \sqrt{T}\cos\left\{\frac{\pi}{4\alpha}(2fT-(1-\alpha))\right\} & \text{si} \quad \frac{1-\alpha}{2T} < |f| < \frac{1+\alpha}{2T} \\ 0 & \text{si} \quad |f| \geq \frac{1+\alpha}{2T} \end{cases} \tag{5.69}$$

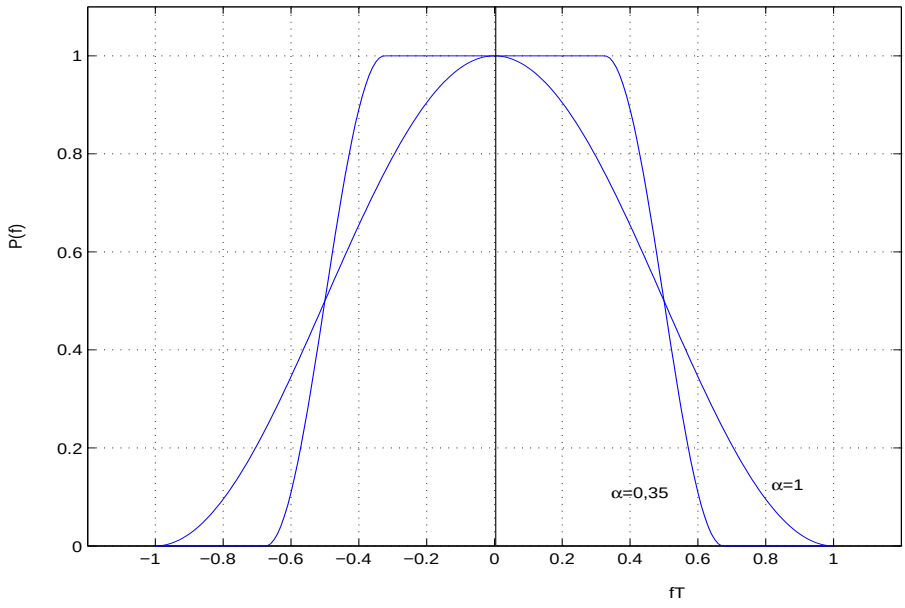


FIG. 5.39 – réponse en fréquence du filtre en cosinus surélevé

Chapitre 6

Introduction aux modulations numériques

L'opération de modulation consiste à adapter le signal à émettre au canal de transmission. Elle a pour effet de translater le spectre du signal autour d'une fréquence porteuse.

6.1 Modulation à déplacement d'amplitude

La modulation à déplacement d'amplitude (MDA) ou *pulse amplitude modulation* (PAM) en anglais établit une correspondance entre les symboles a_k et le signal modulé comme suit :

$$\begin{aligned} x(t) &= \sum_{k=-\infty}^{\infty} a_k p(t-kT) \\ &= \sum_{k=-\infty}^{\infty} a_k g(t-kT) cos(\omega_0 t) \end{aligned} \tag{6.1}$$

$g(t)$ et $p(t) = g(t)cos(\omega_0 t)$ sont les formes d'onde du signal respectivement avant et après la modulation.

Comme pour le code en ligne NRZ multiniveaux, les symboles a_k prennent leurs valeurs dans l'alphabet $\{\pm d, \pm 3d, \ldots, \pm(M-1)d\}$.

Le signal $x(t)$ peut également s'écrire sous la forme suivante :

$$x(t) = \sum_{k=-\infty}^{\infty} \Re\left\{ a_k g(t-kT) e^{j\omega_0 t} \right\} \tag{6.2}$$

Une représentation géométrique des signaux PAM pour $M = 4$ et $M = 8$ est donnée sur la figure 6.1

Sur le canal gaussien, le taux d'erreurs symbole est donné par :

$$TES = \frac{M-1}{M} erfc\left(\sqrt{\frac{3log_2 M}{M^2-1} \frac{E_b}{N_0}} \right) \tag{6.3}$$

Le codage de Gray consiste à choisir les mots binaires associés à deux points adjacents afin qu'ils ne diffèrent que d'un seul bit. Ainsi, une erreur symbole entre deux points adjacents n'engendre qu'une erreur bit sur les $log_2 M$ bits du mot binaire.

exemple : $M = 4$: (00 - 01), (01 - 11) et (11 - 10).

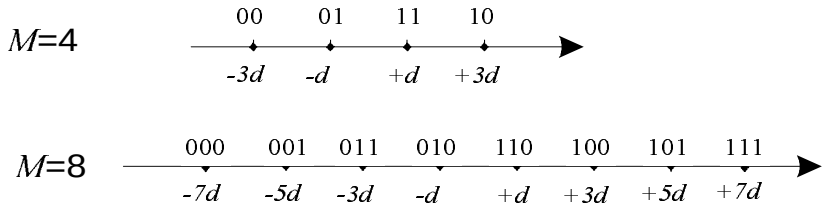


FIG. 6.1 – constellation d'une modulation PAM pour $M = 4$ et $M = 8$.

Si on utilise un codage de Gray, on a donc la relation suivante entre le taux d'erreurs bit et le taux d'erreurs symbole :

$$TEB = \frac{TES}{log_2 M} \tag{6.4}$$

6.2 Modulation à déplacement de phase

La modulation à déplacement de phase (MDP) ou *phase shift keying* (PSK) en anglais consiste à moduler la phase de la porteuse ϕ_k par le symbole à transmettre b_k :

$$\begin{aligned} x(t) &= \sum_{k=-\infty}^{\infty} Ag(t-kT)cos\left(\omega_0 t + \frac{2\pi b_k}{M} + \phi_0\right) \\ &= \sum_{k=-\infty}^{\infty} Ag(t-kT)cos\left(\omega_0 t + \phi_k + \phi_0\right) \end{aligned} \tag{6.5}$$

$b_k \in \{0, 1, \ldots, M-1\}$ et $g(t)$ est la forme d'onde du signal en bande de base. ϕ_0 est une phase de référence. On choisit en général $\phi_0 = \pi/M$.

exemple : $M = 4$: la phase ϕ_k peut prendre les valeurs suivantes : $0, \frac{\pi}{2}, \pi$ et $\frac{3\pi}{2}$.

Le signal $x(t)$ peut également s'écrire sous la forme suivante :

$$x(t) = \sum_{k=-\infty}^{\infty} \Re\left\{a_k g(t-kT)e^{j\omega_0 t}\right\} \tag{6.6}$$

avec

$$a_k = A\exp\left\{j\left(\frac{2\pi b_k}{M} + \phi_0\right)\right\}$$

Une représentation géométrique des signaux PSK pour $M = 4$ (QPSK) et $M = 8$ (8PSK) est donnée sur la figure 6.2

6.3 Modulation d'amplitude de deux porteuses en quadrature

La modulation d'amplitude de deux porteuses en quadrature(MAQ) ou *quadrature amplitude modulation* (QAM) en anglais consiste à moduler simultanément l'amplitude et la phase de la porteuse par le symbole à transmettre a_k :

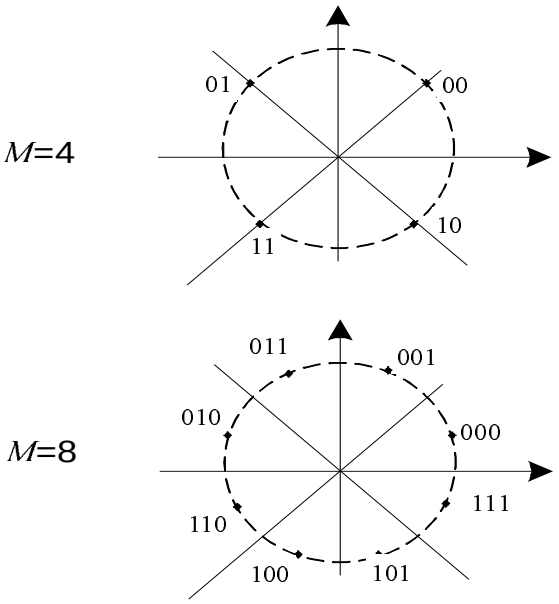


FIG. 6.2 – constellation d'une modulation PSK pour $M = 4$ et $M = 8$.
().

$$
\begin{aligned}
x(t) &= \sum_{k=-\infty}^{\infty} \Re\left\{ a_k g(t-kT) e^{j\omega_0 t} \right\} \\
&= \sum_{k=-\infty}^{\infty} \Re\left\{ (a_k^R + j a_k^I) g(t-kT) e^{j\omega_0 t} \right\} \\
&= \sum_{k=-\infty}^{\infty} a_k^R g(t-kT) cos\omega_0 t - a_k^I g(t-kT) sin\omega_0 t \\
&= \sum_{k=-\infty}^{\infty} v_k g(t-kT) cos(\omega_0 t + \phi_k) \qquad (6.7)
\end{aligned}
$$

où

$$v_k = \sqrt{(a_k^R)^2 + (a_k^I)^2} \qquad \text{et} \qquad \phi_k = \arctan \frac{a_k^I}{a_k^R}$$

Cette modulation permet de réaliser des transmissions numériques d'efficacité spectrale élevée.

Une représentation géométrique des signaux QAM pour $M = 4$ $M = 16$ est donnée sur la figure 6.3

Nous pouvons observer que la modulation $QAM4$ est identique à la modulation à déplacement de phase à 4 états de phase QPSK.

Nous allons maintenant déterminer le taux d'erreurs symbole d'une modulation QAM. Soit $k = log_2 M$ le nombre de bits par symbole.

Pour les constellations rectangulaires ($M = 2^k$), avec k pair, la modulation QAM est équivalente à deux modulations PAM en quadrature ayant chacune $\sqrt{M} = 2^{k/2}$ points. Comme les signaux en phase et en quadrature peuvent être parfaitement séparés à la démodulation, le taux d'erreurs symbole TES d'une modulation QAM peut s'obtenir aisément à partir du TES de la modulation PAM composée de $\sqrt{M}$ points. Alors le TES de la modulation QAM s'exprime comme suit :

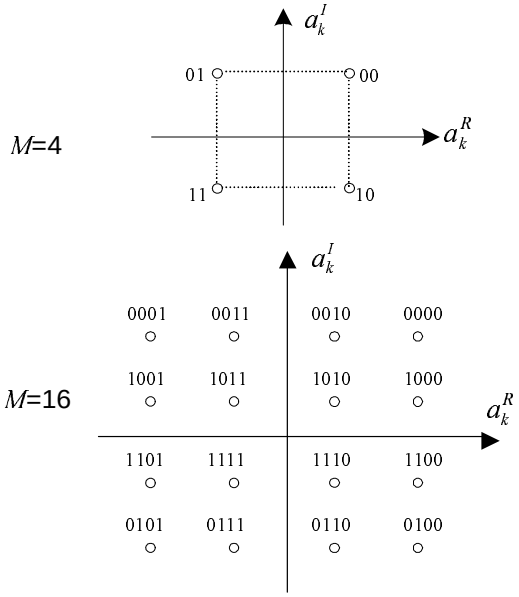


FIG. 6.3 – constellation d'une modulation QAM4 et QAM16.

$$TES = 1 - (1 - TES_{PAM\sqrt{M}})^2 \tag{6.8}$$

où $TES_{PAM\sqrt{M}}$ est le taux d'erreurs symbole de la modulation PAM avec la moitié de la puissance moyenne d'une modulation QAM équivalente :

$$TES_{PAM\sqrt{M}} = \left(1 - \frac{1}{\sqrt{M}}\right) erfc\left(\sqrt{\frac{3log_2M}{2(M-1)}\frac{E_b}{N_0}}\right) \tag{6.9}$$

La relation (6.8) peux alors s'écrire comme suit :

$$TES = 2\left(1 - \frac{1}{\sqrt{M}}\right) erfc\left(\sqrt{\frac{3log_2M}{2(M-1)}\frac{E_b}{N_0}}\right)\left[1 - \frac{1}{2}\left(1 - \frac{1}{\sqrt{M}}\right) erfc\left(\sqrt{\frac{3log_2M}{2(M-1)}\frac{E_b}{N_0}}\right)\right] \tag{6.10}$$

Pour la modulation QAM 4, on obtient par exemple :

$$TES = 1 - \left(1 - \frac{1}{2}erfc\left(\sqrt{\frac{E_b}{N_0}}\right)\right)^2 = erfc\left(\sqrt{\frac{E_b}{N_0}}\right) - \frac{1}{4}erfc\left(\sqrt{\frac{E_b}{N_0}}\right)^2 \tag{6.11}$$

Par ailleurs avec un codage de Gray, le taux d'erreurs bit pour la modulation QAM 4 est égal à :

$$TEB = \frac{1}{2}erfc\left(\sqrt{\frac{E_b}{N_0}}\right) \tag{6.12}$$

En négligeant le second terme de (6.11), on retrouve la relation $TEB = TES/log_2M = TES/2$

Chapitre 7

Introduction au codage de canal

Communication links transmit information from here to there. Computer memories transmit information from now to then.
E.R. Berlekamp

7.1 Introduction

Dans ce cours, nous allons déterminer les limites de communication sans erreur dans un canal bruité. Nous introduirons tout d'abord différents modèles de canaux de transmission. Puis nous nous interesserons à la capacité de ces différents canaux de transmission et au théorème du codage de canal.

7.2 Modèle de canaux de transmission

7.2.1 Le canal binaire symétrique

Le canal binaire symétrique est le canal le plus simple possible puisque l'entrée et la sortie du canal sont binaires.

Ce canal est représenté par le graphe suivant :

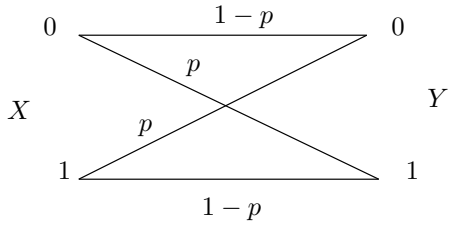


Fig. 7.1 – canal binaire symétrique

Ce canal est caractérisé par les 2 probabilités de transition suivantes :

$$\begin{aligned} P(Y=0|X=1) &= P(Y=1|X=0) = p \\ P(Y=0|X=0) &= P(Y=1|X=1) = 1-p \end{aligned} \tag{7.1}$$

p est appelé la probabilité d'inversion.

$P(X)$	
$X=0$	q
$X=1$	$1-q$

$P(Y\|X)$	$X=0$	$X=1$
$Y=0$	$1-p$	p
$Y=1$	p	$1-p$

On définit également les probabilités a priori $P(X=0)=q$ et $P(X=1)=1-q$.
En utilisant la loi de Bayes, on peut calculer $P(X,Y)$ et $P(Y)$:

$P(X,Y)$	$Y=0$	$Y=1$
$X=0$	$q(1-p)$	qp
$X=1$	$(1-q)p$	$(1-q)(1-p)$

$P(Y)$	
$Y=0$	$q(1-p)+(1-q)p$
$Y=1$	$qp+(1-q)(1-p)$

Le canal binaire symétrique est sans mémoire : soit $\mathbf{x}$ et $\mathbf{y}$ respectivement les séquences d'entrée et de sortie composées de n bits du canal : $\mathbf{x}=[x_1,x_2,\ldots,x_n]$, et $\mathbf{y}=[y_1,y_2,\ldots,y_n]$. La relation suivante justifie l'absence de mémoire dans le canal :

$$P(Y_1=y_1,\ldots,Y_n=y_n|X_1=x_1,\ldots,X_n=x_n)=\prod_{i=1}^{n}P(Y=y_i|X=x_i)$$

La probilité conditionnelle jointe est le produit des n probabilités conditionnelles $P(Y=y_i|X=x_i)$.

L'entropie de la source est :

$$H(X)=-q\log_2 q-(1-q)\log_2(1-q) \tag{7.2}$$

Calculons H(X|Y) :

$$\begin{aligned}H(X|Y)=&-P(X=0,Y=0)\log_2(P(X=0|Y=0))\\&-P(X=0,Y=1)\log_2(P(X=0|Y=1))\\&-P(X=1,Y=0)\log_2(P(X=1|Y=0))\\&-P(X=1,Y=1)\log_2(P(X=1|Y=1))\end{aligned} \tag{7.3}$$

Sur la figure 7.2 nous présentons les courbes $H(X|Y)=f(q)$ pour un canal binaire symétrique avec p=0.1, 0.2 et 0.5.

Si $q=0,5$ alors on a :

$$H(X|Y)=-p\log_2(p)-(1-p)\log_2(1-p) \tag{7.4}$$

7.2.2 Canaux discrets sans mémoire

Le canal binaire symétrique est un cas particulier de la famille des canaux discrets sans mémoire. Les symboles en entrée de ce canal sont M-aire et les symboles en sortie sont N-aire. Il est décrit par un ensemble de NM probabilités conditionnelles de la forme $P(Y=y_j|X=x_i)\equiv p_{ij}$. Ce canal est décrit par le graphe de la figure 7.3.

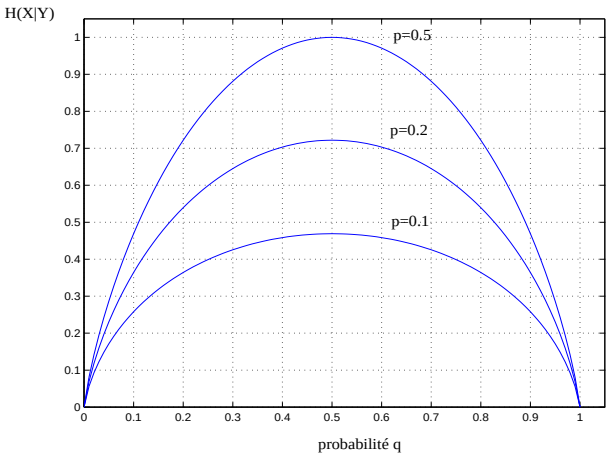


FIG. 7.2 – entropie conditionnelle $H(X|Y)$ en fonction de q et p

7.2.3 Canal à bruit blanc additif gaussien

Le canal à bruit blanc additif gaussien est le canal à alphabet de sortie continu le plus important. Il permet de modéliser les canaux dont le bruit prédominant est le bruit thermique. Nous considérerons que la bande occupée par le signal en entrée du canal est en bande de base limitée à B et que le bruit additif est stationnaire, blanc, gaussien et de densité spectrale de puissance unilatérale N_0. La puissance du bruit N est égale à :

$$N = N_0 B \tag{7.5}$$

A la réception, le démodulateur optimal comprend un filtre adapté limitant la bande de bruit à B.

Le théorème de l'échantillonnage implique que $2BT$ échantillons suffisent pour représenter les signaux en entrée et en sortie du démodulateur pendant une durée T.

Considérons une modulation bipodale où les symboles émis x_i à l'instant i peuvent prendre les amplitudes les valeurs $+\sqrt{E_b}$ ou $-\sqrt{E_b}$.

A l'instant i, on peut exprimer la sortie y_i du démodulateur optimal comme suit :

$$y_i = x_i + n_i \tag{7.6}$$

n_i est l'échantillon réel de bruit blanc centré dont la densité de probabilité est gaussienne :

$$p(n_i) = \frac{1}{\sqrt{\pi N_0}} \exp\left(-\frac{n_i^2}{N_0}\right) \tag{7.7}$$

La variance de l'échantillon de bruit n_i est égale à :

$$\sigma^2 = \frac{N}{2B} = \frac{N_0}{2}$$

Ainsi la densité de probabilité de y_i conditionnellement à x_i est :

$$p(y_i|x_i) = \frac{1}{\sqrt{\pi N_0}} \exp\left\{-\frac{[y_i - x_i]^2}{N_0}\right\} \tag{7.8}$$

7.3 Capacité d'un canal de transmission

7.3.1 Introduction

Cherchons à résoudre le problème suivant : supposons que l'on transmet 1000 bits par seconde (bits équiprobables $P(X = 0) = P(X = 1) = \frac{1}{2}$) dans un canal binaire symétrique de paramètre

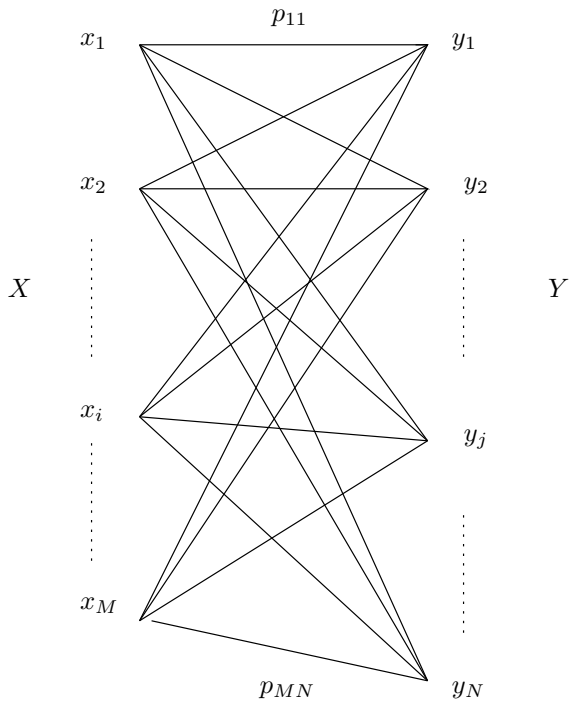


FIG. 7.3 – canal discret sans mémoire

$p = 0.1$. Quel est le débit d'information maximum qu'il est possible de transmettre ? on pourrait penser que ce débit est égal à 900 Sh/s en soustrayant le nombre d'erreurs par seconde. Cependant cette vue simpliste est erronée car nous ne connaissons pas la position de ces erreurs. Par exemple, dans le cas extrême où $p = 0.5$, nous avons en moyenne 500 erreurs par seconde et aucune information n'est transmise !

Essayons maintenant de répondre à cette question.

7.3.2 Capacité d'un canal de transmission

Nous noterons X et Y les variables aléatoires associées respectivement à l'entrée et à la sortie du canal.

définition 1 : On définit la *capacité* d'un canal de transmission par :

$$C = \max I(X, Y) \tag{7.9}$$

Ainsi, la capacité d'un canal de transmission est le maximum de l'information mutuelle moyenne.

Nous verrons plus tard que la capacité telle qu'elle est définie ci-dessus est égale au plus grand nombre de bits d'information qui peuvent être transmis sur le canal avec un taux d'erreurs aussi faible que possible.

Considérons tout d'abord le cas où le canal de transmission est dépourvu de bruit. Dans ce cas, sa capacité est la quantité d'information moyenne maximale que peut transporter chacun des symboles en entrée du canal.

La capacité C s'exprime en Shannon/symbole . Il est également possible de l'exprimer en Shannon/seconde (on parle alors de capacité par unité de temps). Pour la distinguer de la capacité

par symbole, nous noterons cette dernière C'. On a :

$$C' = C \times D_s \quad \text{avec} \quad D_s \quad \text{débit symbole de la source} \tag{7.10}$$

En absence de bruit, la capacité C du canal est égale à $\log_2 Q$. En effet, la quantité d'information moyenne maximale s'obtient lorsque l'entropie de la source est maximale soit lorsque les symboles Q-aire de la source sont équiprobables. On a alors :

$$C = H_{MAX}(X) = \log_2 Q \tag{7.11}$$

En présence de bruit, on a $C < H_{MAX}(X)$. Pour calculer la capacité du canal de transmission, il faut déterminer la quantité d'information moyenne perdue dans celui-ci. Nous avons vu précédemment que $H(X|Y)$ mesure l'incertitude résiduelle sur X connaissant Y. Pour une bonne communication il est souhaitable que cette quantité soit nulle ou négligeable.

La maximisation est réalisée sur l'ensemble de toutes les sources possibles. Si le canal est sans mémoire, cette maximisation est effectuée sur l'ensemble des probabilités p_i d'apparition des symboles d'entrée.

$H(X|Y)$ correspond à la quantité d'information moyenne perdue dans le canal.

Lorsque le canal est sans bruit, on a $H(X|Y) = H(X|X) = 0$ et donc $C = H_{MAX}(X)$. On retrouve la relation (7.11).

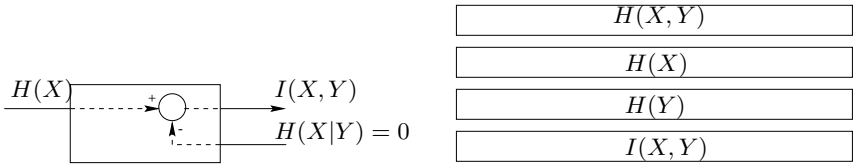


FIG. 7.4 – cas $C = H_{MAX}(X)$

Lorsque le canal est tellement bruyant que X et Y sont indépendants, on a $H(X|Y) = H(X)$. Dans ce cas particulier la capacité du canal de transmission est nulle $C = 0$.

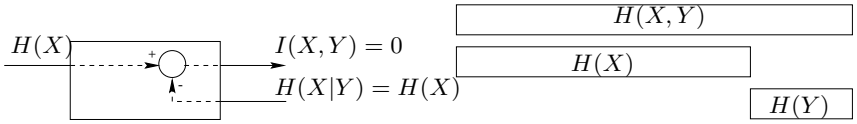


FIG. 7.5 – cas $C = 0$

Lorsque l'entropie de la source est égale à $H_{MAX}(X)$, $H(X|Y)$ ne dépend plus que du canal de transmission utilisé. Si $H(X|Y)$ est non négligeable (cas du canal bruyant), il ne sera pas possible d'effectuer une communication sans erreur en reliant directement la source au canal de transmission. Il faut donc placer un élément appelé codeur de canal entre la source et le canal de transmission. La figure 7.6 présente le nouveau système de communication comprenant un codeur de canal, le canal bruité et un décodeur de canal.

Pour ce nouveau système, on peut définir l'information mutuelle moyenne $I(U,V) = H(U) - H(U|V)$. Le rôle du codage de canal est de rendre la quantité d'information moyenne $H(U|V)$ aussi faible que souhaité. Il est alors possible de transmettre au travers de ce canal bruité une quantité d'information moyenne $H(U)$ avec le critère de qualité souhaité. Bien entendu, on a $H(U) < H(X)$ à cause de la redondance apportée par le codage de canal.

Nous allons maintenant énoncer le théorème fondamental du codage de canal.

FIG. 7.6 – système de communication avec codage de canal

7.3.3 Théorème fondamental du codage de canal

Il existe un codage de canal permettant de garantir une communication avec un taux d'erreurs aussi faible que souhaité à la condition que la quantité d'information moyenne entrant dans l'ensemble codeur-canal-décodeur soit inférieure à la capacité C du canal [27] :

$$H(U) < C \tag{7.12}$$

En multipliant les deux termes de cette inégalité par D_s le débit de la source on obtient l'inégalité entre le débit maximum d'information binaire D_b et la capacité par unité de temps :

$$D_b < C' \tag{7.13}$$

La démonstration de ce théorème est basée sur le principe du codage aléatoire. Les travaux de Shannon ne donnaient pas de solution pour réaliser pratiquement un codeur et un décodeur. Depuis 1948, les chercheurs ont proposé des codes correcteurs d'erreurs et des algorithmes de décodage de complexité raisonnable permettant de s'approcher de cette limite théorique. Ce n'est qu'en 1993 avec la découverte des Turbo codes [3] et en 1995 avec la redécouverte des codes LDPC [?] qu'il a enfin été possible de s'approcher à moins de 1 dB de cette limite.

7.3.4 Capacité d'un canal binaire symétrique

Le canal binaire symétrique est simplement décrit par la probabilité d'erreur p.

Sur la figure 7.7 nous présentons les courbes $I(X,Y) = f(q)$ pour un canal binaire symétrique avec p=0.0,0.1, 0.2 et 0.5.

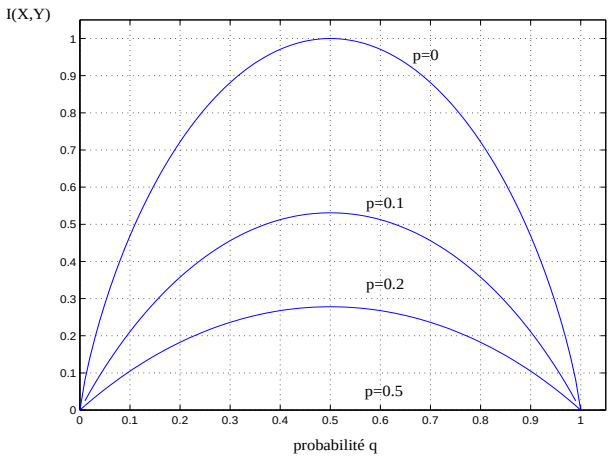


FIG. 7.7 – information mutuelle $I(X,Y)$ en fonction de q et p

Nous pouvons voir sur cette figure que l'information mutuelle est maximisée lorsque les probabilités d'entrée sont identiques : $P(X=0) = P(X=1) = q = 1/2$.

Sur la figure 7.8 nous présentons la courbe $I(X,Y) = f(p)$ pour un canal binaire symétrique avec q=0.5. On a alors :

$$I(X,Y) = H(X) - H(X|Y) = 1 + p\log_2(p) + (1-p)\log_2(1-p) \tag{7.14}$$

Comme attendu, l'information mutuelle est maximale lorsque $p = 0$. Ainsi la capacité du canal binaire symétrique est égale à 1 Shannon/symbole.

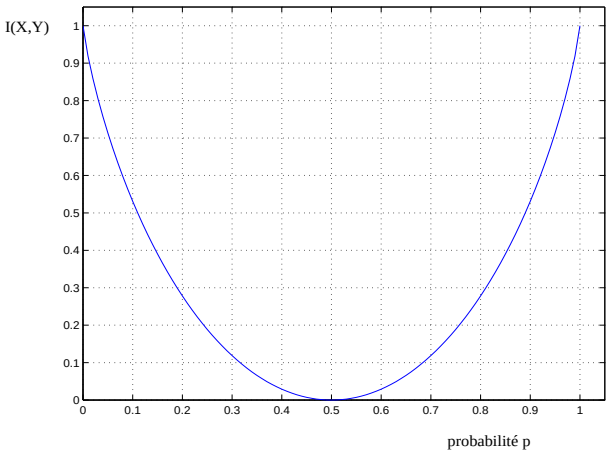


FIG. 7.8 – information mutuelle $I(X,Y)$ en fonction de p pour $q = 0.5$

7.3.5 Capacité d'un canal à bruit blanc additif gaussien

Pour déterminer la capacité d'un canal à bruit blanc additif gaussien, nous allons tout d'abord calculer l'information mutuelle moyenne $I(X,Y)$ bien que les variables aléatoires X et Y soient continues. Pour ce faire, introduisons l'entropie différentielle $H_D(V)$ d'une variable aléatoire V :

$$H_D(V) = -\int_{-\infty}^{+\infty} p(v)\log_2 p(v)dv \tag{7.15}$$

L'information mutuelle peut s'exprimer alors ainsi :

$$\begin{aligned} I(X,Y) &= H_D(Y) - H_D(Y|X) \\ &= H_D(Y) - H_D(X+Z|X) \\ &= H_D(Y) - H_D(Z|X) \\ &= H_D(Y) - H_D(Z) \end{aligned} \tag{7.16}$$

car Z la variable aléatoire associée au bruit est indépendante de X.

Calculons l'entropie différentielle $H_D(V)$ de V, variable aléatoire gaussienne centrée de variance σ_v^2. On a :

$$H_D(V) = -\frac{1}{\sqrt{2\pi}\sigma_v}\int_{-\infty}^{+\infty}\exp\left(\frac{-v^2}{2\sigma_v^2}\right)\log_2\left[\frac{1}{\sqrt{2\pi}\sigma_v}\exp\left(\frac{-v^2}{2\sigma_v^2}\right)\right]dv \tag{7.17}$$

$$= -\frac{1}{\sqrt{2\pi}\sigma_v}\int_{-\infty}^{+\infty}\exp\left(\frac{-v^2}{2\sigma_v^2}\right)\left[\log_2\left(\frac{1}{\sqrt{2\pi}\sigma_v}\right) - \frac{v^2}{2\sigma_v^2\ln 2}\right]dv \tag{7.18}$$

Puisque $\log_2\left(\frac{1}{\sqrt{2\pi}\sigma_v}\right)$ ne dépend pas de v et que

$$\frac{1}{\sqrt{2\pi}\sigma_v}\int_{-\infty}^{+\infty}\exp\left(\frac{-v^2}{2\sigma_v^2}\right)=1$$

on peut extraire le premier membre sous l'intégrale. On obtient alors :

$$H_D(V)=\log_2\left(\sqrt{2\pi}\sigma_v\right)+\frac{1}{\sqrt{2\pi}\sigma_v}\int_{-\infty}^{+\infty}\frac{v^2}{2\sigma_v^2\ln 2}\exp\left(\frac{-v^2}{2\sigma_v^2}\right)dv \tag{7.19}$$

Par définition,

$$E(V^2)=\frac{1}{\sqrt{2\pi}\sigma_v}\int_{-\infty}^{+\infty}v^2\exp\left(\frac{-v^2}{2\sigma_v^2}\right)dv=\sigma_v^2 \tag{7.20}$$

Finalement, on obtient donc :

$$H_D(V)=\frac{1}{2\ln 2}+\log_2\left(\sqrt{2\pi}\sigma_v\right) \tag{7.21}$$

On peut montrer que le maximum de $I(X,Y)$ s'obtient lorsque la densité de probabilité de X est gaussienne, centrée et de variance σ_x^2. La variance du bruit Z est égale à $\sigma_n^2=\frac{N_0}{2}$ où N_0 est la densité spectrale unilatérale du bruit blanc gaussien.

Calculons la variance de Y :

$$E(Y^2)=E(X+Z)^2=E(X)^2+2E(XZ)+E(Z)^2=E(X)^2+E(Z)^2=\sigma_x^2+\frac{N_0}{2} \tag{7.22}$$

A partir de (7.16) on dérive la capacité du canal BBAG :

$$\begin{aligned}C&=\max I(X,Y)\\&=\log_2\sqrt{\pi(2\sigma_x^2+N_0)}-\log_2\sqrt{\pi N_0}\\&=\frac{1}{2}\log_2\left(1+\frac{2\sigma_x^2}{N_0}\right)\qquad\text{en Shannon/symbole}\end{aligned} \tag{7.23}$$

Introduisons P la puissance moyenne du signal X. En considérant une fréquence d'échantillonage égale à $2B$, nous avons $2BT$ échantillons pendant une durée T. On a alors :

$$\begin{aligned}P&=\frac{1}{T}\int_0^T E[x^2(t)]dt\\&=\frac{1}{T}\sum_{i=1}^{2BT}E(x_i^2)\\&=\frac{1}{T}\sum_{i=1}^{2BT}\sigma_x^2\\&=2B\sigma_x^2\end{aligned} \tag{7.24}$$

On obtient finalement à partir (7.23) de l'expression classique de la capacité d'un canal BBAG :

$$\boxed{C=\frac{1}{2}\log_2\left(1+\frac{P}{N}\right)\qquad\text{en Shannon/symb}} \tag{7.25}$$

La capacité C est une capacité par symbole c'est-à-dire par échantillon du signal. Certains auteurs parlent de capacité par dimension souvent exprimée en bit/dimension.

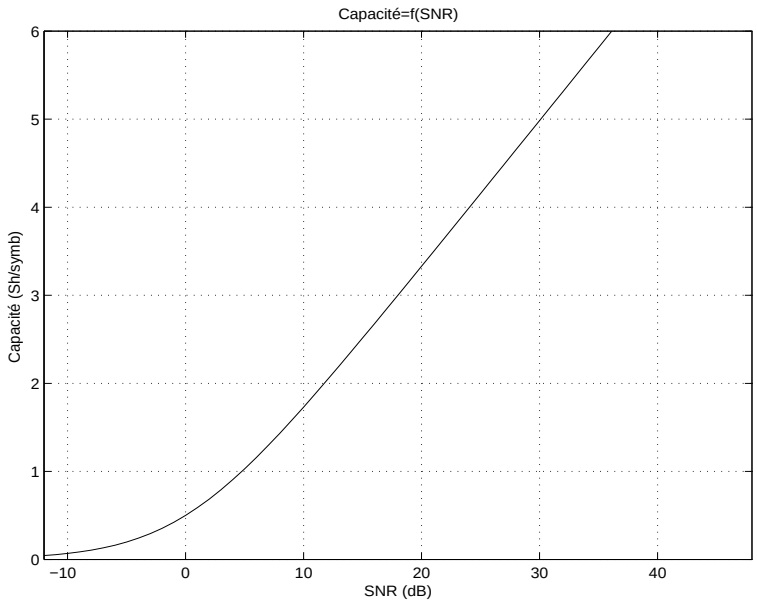


FIG. 7.9 – Capacité d'un canal à bruit blanc additif gaussien.

La figure (7.9) présente la courbe de la capacité C du canal BBAG en fonction du rapport signal à bruit $SNR = \frac{P}{N}$. On peut observer que pour $SNR > 5dB$, la capacité C est bien approximée par la fonction linéaire $C \approx \frac{1}{2}\log_2(SNR)$.

En multipliant (7.25) par la fréquence d'échantillonage $2B$, on obtient finalement l'expression de la capacité par unité de temps suivante :

$$C' = B\log_2\left(1+\frac{P}{N}\right) \qquad \text{en Shannon/sec} \qquad (7.26)$$

avec N puissance du bruit $N = BN_0$

Lorsque le rapport signal à bruit est grand, on peut approximer la capacité du canal BBAG comme suit :

$$C' = B\log_2\left(1+\frac{P}{N}\right) \approx B\frac{\log_{10}(P/N)}{\log_{10} 2} \approx \frac{B}{3}(P/N)_{dB}$$

7.3.6 Représentation géométrique

Il est également possible de démontrer géométriquement que pour garantir une transmission sans erreur, la quantité d'information moyenne $H(U)$ ne peut pas être supérieure à $\frac{1}{2}\log_2\left(1+\frac{2\sigma_x^2}{N_0}\right)$

On rappelle la relation vectorielle entre le vecteur émis $\mathbf{x}$ et le vecteur reçu $\mathbf{y}$ de dimension D

$$\mathbf{y} = \mathbf{x} + \mathbf{n} \qquad (7.27)$$

$\mathbf{n} = (n_1, n_2, \ldots, n_D)$ est le vecteur bruit composé de D composantes indépendantes gaussiennes de variance $\sigma_n^2 = \frac{N_0}{2}$. La densité de probabilité du vecteur $\mathbf{n}$ s'exprime comme suit :

$$p(\mathbf{n}) = \frac{1}{(2\pi\sigma_n^2)^{D/2}}\exp\left(-\frac{\sum_{i=1}^{D} n_i^2}{2\sigma_n^2}\right) \qquad (7.28)$$

Pour D tendant vers l'infini, nous avons montré dans l'annexe 1 que la norme du vecteur de bruit [1] est concentrée à la surface de la sphère à D dimensions de rayon $\sqrt{D\sigma_n^2}$.

Le vecteur émis $\mathbf{x}$ est généré aléatoirement avec une variance σ_x^2 et une distribution gaussienne afin de maximiser la capacité :

$$p(\mathbf{x}) = \frac{1}{(2\pi\sigma_x^2)^{D/2}} \exp\left(-\frac{\sum_{i=1}^{D} x_i^2}{2\sigma_x^2}\right) \tag{7.29}$$

Pour la même raison que précédemment, la norme du vecteur $\mathbf{x}$ est concentrée à la surface de la sphère de rayon $\sqrt{D.\sigma_x^2}$. Comme la puissance du signal reçu est égale à la somme $\sigma_x^2 + \sigma_n^2$, le vecteur correspondant au signal reçu quelque soit le signal émis se trouve à la surface de la sphère à D dimension de rayon $\sqrt{D(\sigma_x^2 + \sigma_n^2)}$.

On souhaite réaliser une transmission sans erreur d'une quantité d'information moyenne $H(U) = \frac{1}{D}\log_2 M$, où $M = 2^{DH(U)}$ est le nombre de signaux émis possibles. Pour ce faire, il faut que les sphères de bruit soient disjointes. Ainsi le volume des M sphères de bruit doit être inférieur au volume de la sphère de rayon $\sqrt{D(\sigma_x^2 + \sigma_n^2)}$. Rappelons que le volume d'une sphère à D dimension et de rayon r est donné par

$$V(r, D) = \frac{\pi^{D/2}}{\Gamma(D/2+1)} r^D \tag{7.30}$$

où $\Gamma(.)$ est la fonction factorielle d'Euler [2]

Ainsi, on doit avoir :

$$\text{nombre de signaux distinguables} = M \leq \frac{V(\sqrt{D(\sigma_x^2 + \sigma_n^2)}, D)}{V(\sqrt{D.\sigma_n^2}, D)} \tag{7.31}$$

Cette idée est illustrée sur la figure 7.10.

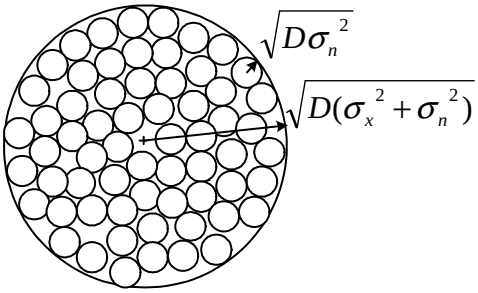


FIG. 7.10 – Répartition des sphères de bruit.

L'expression se simplifie comme suit :

$$M \leq \frac{(D(\sigma_x^2 + \sigma_n^2))^{D/2}}{(D.\sigma_n^2)^{D/2}} \tag{7.32}$$

On obtient l'inégalité suivante :

$$M \leq \left(\frac{\sigma_x^2 + \sigma_n^2}{\sigma_n^2}\right)^{D/2} \tag{7.33}$$

[1] norme du vecteur de bruit $= \sqrt{\sum_{i=1}^{D} n_i^2}$

[2] la fonction factorielle d'Euler $\Gamma(.)$ est définie comme suit :
$\Gamma(n) = (n-1)!$ avec $n \in \mathbb{N}^*$
$\Gamma(n+1/2) = \frac{(2n)!}{2^{2n}.n!}.\sqrt{\pi}$ avec $n \in \mathbb{N}^*$
$\Gamma(1/2) = \sqrt{\pi}$

Cette inégalité s'écrit alors :

$$H(U) \leq \frac{1}{2}\log_2\left(1+\frac{\sigma_x^2}{\sigma_n^2}\right) \tag{7.34}$$

Finalement comme la capacité C est la plus grande valeur que peut prendre la quantité d'information moyenne $H(U)$, on retrouve la formule de la capacité du canal BBAG :

$$C = \frac{1}{2}\log_2\left(1+\frac{\sigma_x^2}{\sigma_n^2}\right) \qquad \text{Sh/dim} \tag{7.35}$$

Lorsque la bande passante est limitée, la dimension D est égale à $D = 2BT$ avec B bande passante du système et T durée de la transmission. La puissance du bruit est égale à $N = N_0B = 2B\sigma_n^2$ et la puissance moyenne du signal X est égale à $P = 2B\sigma_x^2$. Alors la capacité C' par unité de temps est :

$$\begin{aligned} C' &= B\log_2\left(1+\frac{P}{N}\right) \\ &= B\log_2\left(1+\frac{P}{N_0B}\right) \qquad \text{en Shannon/sec} \end{aligned} \tag{7.36}$$

Exemple : Considérons la ligne téléphonique dont le rapport signal à bruit P/N est de l'ordre de 30 dB et la bande passante est limitée à B= 3.5kHz. La capacité est d'environ 35 Kb/s. Il devrait donc être possible en utilisant des codes correcteurs et des modulations performantes de transmettre sur cette ligne jusqu'à 35 Kb/s sans erreur de transmission. La norme V.34 sur les modems permet d'atteindre 33.6 Kb/s ce qui est proche de la limite théorique.

Soit E_b l'énergie moyenne par bit d'information et E_s l'énergie moyenne par symbole. On a :

$$P = \frac{E_s}{T_s} = \frac{E_b}{T_b} \tag{7.37}$$

T_s et T_b sont respectivement la durée de transmission d'un symbole et d'un bit d'information (en considérant le cas d'une modulation M-aire $M = 2^g$ et un code de rendement R on a $T_s = gRT_b$).

On a la relation suivante entre le rapport signal à bruit P/N et le rapport E_b/N_0 :

$$\frac{P}{N} = \frac{E_s}{N_0BT_s} = \frac{E_b}{N_0BT_b} = \eta\frac{E_b}{N_0} \tag{7.38}$$

η est l'efficacité spectrale en bits/sec/Hz :

$$\eta = \frac{D_b}{B} \quad \text{avec} \quad D_b = \frac{1}{T_b} \quad \text{débit binaire d'information} \tag{7.39}$$

L'efficacité spectrale maximale η est maximale lorsque la bande passante est minimale soit $B_{min} = 1/T_s$, on a :

$$\eta_{max} = \frac{1}{T_bB_{min}} = \frac{T_s}{T_b} \tag{7.40}$$

En considérant que le débit binaire est égal à la capacité du canal ($D_b = C'$), l'efficacité spectrale η_{max} s'écrit aussi :

$$\eta_{max} = \frac{C'}{B} = \log_2\left(1+\eta_{max}\frac{E_b}{N_0}\right) \qquad \text{en bits/sec/Hz} \tag{7.41}$$

Cette équation peut s'écrire encore :

$$\frac{E_b}{N_0} = \frac{2^{\eta_{max}} - 1}{\eta_{max}} \tag{7.42}$$

La valeur minimale de E_b/N_0 pour une communication sans erreur s'obtient lorsque l'efficacité spectrale maximale tend vers zéro (la bande passante tend vers l'infini). On obtient :

$$\lim_{\eta_{max} \to 0} \frac{E_b}{N_0} = \ln 2 \quad \text{soit} \quad \left.\frac{E_b}{N_0}\right|_{dB} = -1.59 \quad \text{dB} \tag{7.43}$$

La figure 7.11 présente la courbe de l'efficacité spectrale maximale en fonction du rapport E_b/N_0. Comme exemple, nous avons également donné les rapports E_b/N_0 pour un taux d'erreur bit de 10^{-5} des systèmes utilisant une modulation de phase à 2 et 4 états (MDP2 et MDP4) sans codage. Les performances de ces systèmes de communication sont respectivement à 9.5 dB et 7.75 dB de la limite de Shannon. L'ajout d'un code convolutif (133,171) de rendement $R = 1/2$ à un système utilisant une modulation MDP2 apporte un gain de 5.1 dB par rapport au système sans codage. La concaténation de ce code convolutif avec un code Reed-Solomon (255,223) proposée par Forney [10] permet de s'approcher à 2.5 dB de la limite de Shannon. Le dernier point correspond aux performances obtenues en utilisant les codes convolutifs concaténés en parallèle ou Turbo codes introduits en 1993 par Berrou *et al.* [3]

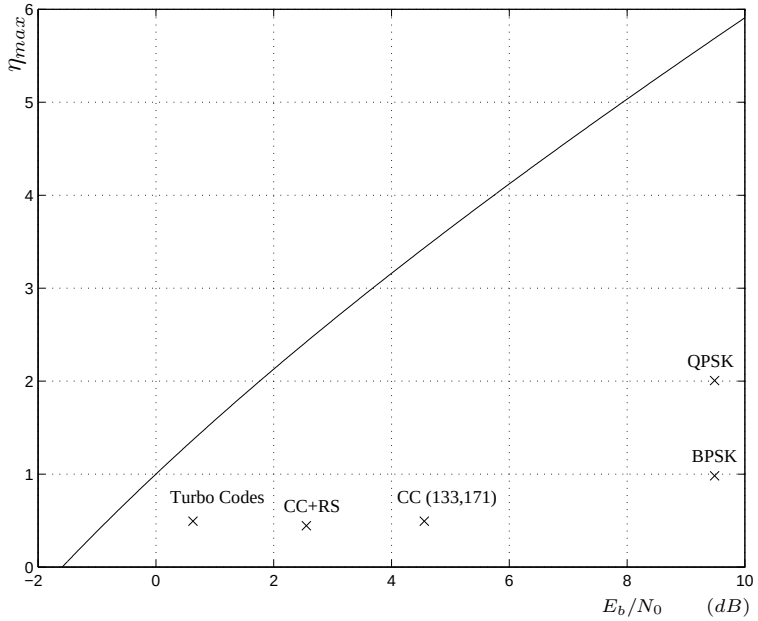


FIG. 7.11 – Efficacité spectrale maximale d'un canal à bruit blanc additif gaussien.

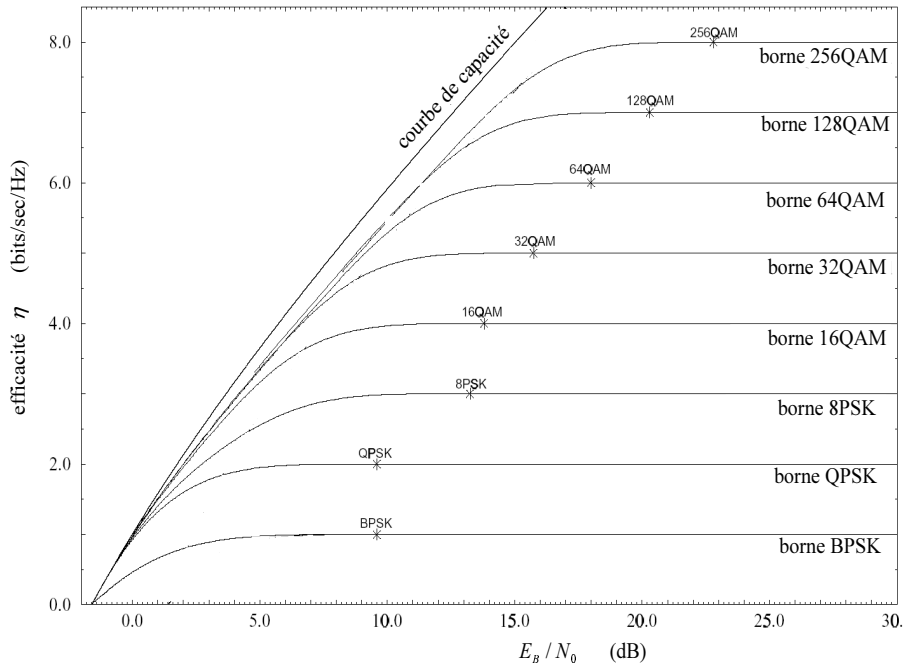


FIG. 7.12 – Efficacité spectrale en fonction du rapport E_b/N_0

Chapitre 8

Codes correcteurs d'erreurs en bloc

8.1 Introduction

Dans ce paragraphe nous étudierons les aspects principaux des codes correcteurs d'erreurs. Pour une étude plus exhaustive, nous renvoyons le lecteur à l'ouvrage de McWilliam et Sloane [21] et aux livres en français de Cohen [7] et Battail [1].

L'objectif du codage de canal est de protéger les données issues du codage de source contre les erreurs de transmission. Une première solution consiste à utiliser un codage aléatoire puisque nous avons vu précédemment que ce codage permet d'atteindre la limite du théorème du codage de canal lorsque $N \rightarrow +\infty$. Nous allons cependant voir que cette technique n'est pas envisageable pratiquement.

Soit un code $\mathcal{C}$ en bloc binaire aléatoire (N, K) comprenant 2^K mots de code. Chaque mot d'information composé de K bits est associé à un mot de code unique composé de N bits.

Pour réaliser un codeur aléatoire, il faut tout d'abord construire une liste de 2^K mots de code. Chacun des mots de code est composé de N bits tirés aléatoirement. L'opération de codage consiste à associer à chaque mot d'information une adresse unique qui servira ensuite pour lire le mot de code correspondant.

Le contenu de la liste ne présentant aucune structure particulière, l'opération de décodage implique de faire la comparaison exhaustive du mot reçu en sortie du canal avec l'ensemble des 2^K mots de code avant de déterminer le mot de code le plus probable. La complexité du décodeur croit exponentionellement avec K et rend cette technique presque toujours inutilisable en pratique.

L'impossibilité pratique d'utiliser le codage aléatoire nous amène donc à utiliser des codes possédant une structure algébrique comme par exemple la linéarité et rendant ainsi les opérations de codage et de décodage plus simples à effectuer.

Ces codes doivent de plus être adaptés aux types d'erreurs de transmission (aléatoire, isolé ou par paquets).

Nous nous intéresserons aux trois principales familles de codes suivantes :

-les codes en bloc linéaires

Un code en bloc q-aire (N, K) est un ensemble comprenant q^K mots de code. On associe à chaque mot d'information composé de K symboles q-aire un mot de code composé de N symboles q-aire. Chaque mot de code ne dépend que d'un mot d'information. La linéarité signifie que les N symboles du mot code sont obtenus par combinaison linéaire des K symboles du mot d'information. Les séquences d'information ou d'entrée sont découpées par mot de K symboles q-aire.

-les codes convolutifs

A la différence des codes en bloc, pour un code convolutif de rendement k/n, le mot de code composé de n symboles dépend du mot d'information de k symboles mais aussi d'un nombre fini de mot d'information précédent. Les séquences d'entrée et de sortie sont de longueur infinie.

-les codes concaténés

Ces codes s'obtiennent par concaténation de codes en bloc linéaires ou de codes convolutifs.

8.2 Les corps finis

8.2.1 Rappel sur les corps

Un corps F est un ensemble non vide muni de deux lois de composition internes, l'addition et la multiplication et satisfaisant les axiomes suivants :
1- F est un groupe commutatif par rapport à l'addition (associativité, élément neutre noté 0, symétrique, commutativité)
2- la multiplication est associative : si $a, b, c \in F$, alors $a(bc) = (ab)c$
3- la multiplication est commutative : si $a, b \in F$, alors $ab = ba$
4- la multiplication est distributive à droite et à gauche par rapport à l'addition : si $a, b, c \in F$, alors $a(b + c) = ab + ac$ et $(a + b)c = ac + bc$
5- le corps contient un élément neutre noté 1 pour la multiplication
6- tout élément de F non nul est inversible ; si $a \in F(a \neq 0)$, a^{-1} est son inverse avec $aa^{-1} = 1$

8.2.2 Les corps de Galois

Un corps de Galois est un corps fini possédant q éléments. Il est noté $GF(q)$ (*Galois Field* en anglais) ou $\mathbb{F}_q$. Il est possible de construire un corps de Galois à condition que q soit un nombre premier ou soit de la forme $q = p^m$ avec p nombre premier. Lorsque q est un nombre premier, l'addition et la multiplication dans le corps fini $GF(q)$ correspondent à l'addition et la multiplication modulo q. Tout corps de Galois doit contenir au moins les éléments neutres 0 et 1. Ainsi, le corps de Galois le plus simple est le corps $GF(2)$.

exemple 1 : addition et multiplication dans $GF(2)$

+	0	1
0	0	1
1	1	0

*	0	1
0	0	0
1	0	1

exemple 2 : addition et multiplication dans $GF(5)$

+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

*	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

Un corps de Galois $GF(p^m)$ est isomorphe au corps des polynômes à coefficients dans $GF(p)$ modulo un polynôme irréductible dans $GF(p)$ et de degré m. $GF(p)$ est appelé corps de base.

8.3 Codes en bloc linéaires binaires

Un code $\mathcal{C}$ en bloc linéaire q-aire (N, K) est un ensemble comprenant q^K mots de code. On associe à chaque mot d'information composé de K symboles q-aire un mot de code composé de N symboles q-aire. Un code en bloc est linéaire si les N symboles du mot code sont obtenus par combinaison linéaire des K symboles du mot d'information. Cette propriété permet en particulier de décrire l'opération de codage sous une forme matricielle.

Dans la suite, nous nous intéresserons aux codes en bloc linéaires binaires pour lesquels on a $q = 2$. Il faut préciser qu'un code linéaire en bloc q-aire (N, K) avec $q = 2^p$ peut être vu comme un code en bloc linéaire binaire (pK, pN).

Il est pratique de représenter les mots d'information et les mots de code par des vecteurs. Soit $\mathbf{u} = [u_1, u_2, \ldots, u_K]$ un mot d'information composé de K bits d'information et $\mathbf{c} = [c_1, c_2, \ldots, c_N]$ le mot de code associé composé de N bits. On a la relation matricielle suivante entre le mot d'information $\mathbf{u}$ et le mot de code associé $\mathbf{c}$:

$$\mathbf{c} = \mathbf{uG} \tag{8.1}$$

$\mathbf{G}$ est la matrice génératrice du codeur de dimension $K \times N$.

$$\mathbf{G} = \begin{pmatrix} \mathbf{g}_1 \\ \mathbf{g}_2 \\ \vdots \\ \mathbf{g}_K \end{pmatrix} = \begin{pmatrix} g_{11} & g_{12} & \cdots & g_{1N} \\ g_{21} & g_{22} & \cdots & g_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ g_{K1} & g_{K2} & \cdots & g_{KN} \end{pmatrix} \tag{8.2}$$

Chaque mot de code est une combinaison linéaire des vecteurs $\mathbf{g}_i$ de $\mathbf{G}$. Ainsi donc, un code en bloc linéaire peut être défini comme un sous espace vectoriel à $K < N$ dimensions construit suivant (8.2).

Il est toujours possible en combinant les lignes entre elles de mettre la matrice génératrice G sous la forme systématique suivante :

$$\mathbf{G} = [\mathbf{I_K} \quad \mathbf{P}] = \begin{pmatrix} 1 & 0 & 0 & \ldots & 0 & p_{11} & p_{12} & \cdots & p_{1N-K} \\ 0 & 1 & 0 & \ldots & 0 & p_{21} & p_{22} & \cdots & p_{2N-K} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & 1 & p_{K1} & p_{K2} & \cdots & p_{KN-K} \end{pmatrix} \tag{8.3}$$

Exemple 1 : code de répétition $\mathcal{C}_1$ $(3, 1)$:

$$\mathbf{G} = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix} \tag{8.4}$$

On répète 3 fois chaque bit d'information :

$$c_1 = c_2 = c_3 = u_1$$

Exemple 2 : code de parité $\mathcal{C}_2$ $(3, 2)$:

$$\mathbf{G} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \tag{8.5}$$

c_3 est le bit de contrôle de parité :

$$c_3 = u_1 + u_2$$

Chaque mot de code a un nombre pair de 1. Les mots de code du code $\mathcal{C}_2$ sont 000, 011, 110, 101.

La figure 8.1 donne une représentation graphique de ce code dans un espace à 3 dimensions.

Exemple 3 : code $\mathcal{C}_3$ $(7, 4)$ avec la matrice génératrice suivante :

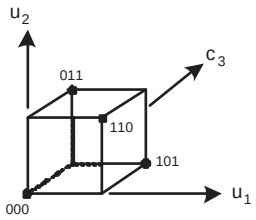


FIG. 8.1 – Code de parité $\mathcal{C}_2$ $(3, 2)$.

$$\mathbf{G} = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} \tag{8.6}$$

Ici, la matrice génératrice sous la forme systématique s'obtient simplement en ajoutant les lignes 3 et 4 à la ligne 1, et la ligne 4 à la ligne 2. Cette technique de combinaison de ligne permet toujours de convertir une matrice génératrice quelconque en une matrice génératrice systématique.

On peut vérifier que la forme systématique de cette matrice génératrice est la suivante :

$$\mathbf{G} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} \tag{8.7}$$

Les trois bits de parité sont obtenus comme suit :

$$\begin{aligned} c_5 &= u_1 + u_2 + u_3 \\ c_6 &= u_2 + u_3 + u_4 \\ c_7 &= u_1 + u_2 + u_4 \end{aligned}$$

8.3.1 Propriétés et définitions

rendement : le rendement R d'un code en bloc (N, K) est égal à :

$$R = \frac{K}{N} \tag{8.8}$$

linéarité : soit $\mathbf{c}_1$ et $\mathbf{c}_2$ deux mots de code du code $\mathcal{C}$, et α_1 et α_2 deux éléments du corps fini. La linéarité implique que $\alpha_1\mathbf{c}_1 + \alpha_2\mathbf{c}_2$ est aussi un mot de code de $\mathcal{C}$. Par conséquence, le mot $\mathbf{c}_0 = [00\ldots0]$ est toujours un mot de code d'un code linéaire. On appellera ce mot de code le mot de code nul.

distance de Hamming : soit $\mathbf{c}_1$ et $\mathbf{c}_2$ deux mots de code du code $\mathcal{C}$ de longueur N, la distance de Hamming $d_H(\mathbf{c}_1, \mathbf{c}_2)$ est égale aux nombres de bits qui différent.

Exemple : $\mathbf{c}_1 = [001100]$ et $\mathbf{c}_2 = [001111]$, $d_H(\mathbf{c}_1, \mathbf{c}_2) = 2$

poids de Hamming : le poids de Hamming $w(\mathbf{c})$ d'un mot de code binaire $\mathbf{c}$ est égal au nombre de bits non nuls de ce mot de code.

Exemple : $\mathbf{c} = [001100]$, $w(\mathbf{c}) = 2$

distance minimale : La distance minimale d_{min} du code $\mathcal{C}$ est le nombre de bits qui différent entre les deux mots de code les plus proches au sens de la distance de Hamming :

$$d_{min} = \min_{i,j,i\neq j} d_H(\mathbf{c}_i, \mathbf{c}_j) \tag{8.9}$$

Lorsque le code est linéaire, la distance minimale d_{min} est égale au poids de Hamming minimal du code $\mathcal{C}$ (en excluant le mot de code nul $\mathbf{c}_0$) :

$$d_{min} = \min_{i,i\neq 0} w(\mathbf{c}_i) \tag{8.10}$$

Jusqu'à récemment, la distance minimale était l'unique critère pour évaluer les performances d'un code correcteur d'erreurs. Ce critère a été critiqué suite à l'avènement de familles de codes très performants imitant le codage aléatoire [3] [1].

Exemple : d_{min}(code $\mathcal{C}_1$)=3 ; d_{min}(code $\mathcal{C}_2$)=2 ; d_{min}(code $\mathcal{C}_3$)=3.

8.3.2 Fonctions d'énumération de poids

Les fonctions d'énumération de poids permettent d'étudier les performances des codes correcteurs d'erreurs en bloc linéaires.

Définition 1 : la fonction d'énumération de poids WEF (*weight enumerator function* en anglais) d'un codeur binaire en bloc systématique (N, K) est définie comme suit :

$$A(D) = \sum_{d=0}^{N} A_d D^d \tag{8.11}$$

A_d est le nombre de mots de code de longueur N de poids d.

Définition 2 : la fonction d'énumération de poids IRWEF (*input redundancy weight enumerator function* en anglais) d'un codeur binaire en bloc systématique (N, K) est définie comme suit :

$$A(W, Z) = \sum_{w=0}^{K} \sum_{z=0}^{N-K} A_{w,z} W^w Z^z \tag{8.12}$$

$A_{w,z}$ est le nombre de mots de code de longueur N dont le poids de la séquence des bits d'information est égal à w et dont le poids de la séquence des bits de redondance est égal à z.

La fonction IRWEF peut aussi s'écrire :

$$A(W, Z) = \sum_{w=0}^{K} A(w, Z) W^w \tag{8.13}$$

avec

$$A(w, Z) = \sum_{z=0}^{N-K} A_{w,z} Z^z \tag{8.14}$$

Exemple : code de parité (3,2)

Les fonctions d'énumération WEF et IRWEF pour le code de parité (3,2) sont donc les suivantes :

$$A(D) = 1 + 3D^2$$
$$A(W, Z) = 1 + 2WZ + W^2$$

TAB. 8.1 – Enumération des poids des mots d'information et des mots de code pour le code de parité (3,2)

u	**x**	w	z	d
00	00 0	0	0	0
01	01 1	1	1	2
10	10 1	1	1	2
11	11 0	2	0	2

8.3.3 Matrice de contrôle

Associé à chaque code $\mathcal{C}$ linéaire en bloc binaire (N, K) il existe un code linéaire en bloc binaire dual $(N, N-K)$. Soit $\mathbf{H}$ la matrice génératrice de ce code dual. Chacun des mots de code $\mathbf{c}$ du code $\mathcal{C}$ est orthogonal à tous les mots de code du code dual :

$$\mathbf{c}\mathbf{H}^T = \mathbf{0}$$

Puisque cette relation est valide pour tous les mots de code du code $\mathcal{C}$, on a la relation entre la matrice génératrice $\mathbf{G}$ du code $\mathcal{C}$ et $\mathbf{H}$:

$$\mathbf{G}\mathbf{H}^T = \mathbf{0}$$

Si la matrice génératrice $\mathbf{G}$ est systématique de la forme (8.3), $\mathbf{H}$ est de la forme suivante :

$$\mathbf{H} = [\mathbf{P}^T \quad \mathbf{I}_{N-K}] = \begin{pmatrix} p_{11} & p_{21} & \cdots & p_{K1} & 1 & 0 & 0 & \cdots & 0 \\ p_{12} & p_{22} & \cdots & p_{K2} & 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ p_{1N-K} & p_{2N-K} & \cdots & p_{KN-K} & 0 & 0 & 0 & \cdots & 1 \end{pmatrix} \tag{8.15}$$

La matrice $\mathbf{H}$ est appelée matrice de contrôle ou matrice de parité du code $\mathcal{C}$

exemple 3 (suite) : la matrice de contrôle du code $\mathcal{C}_3$ $(7, 4)$ est la suivante :

$$\mathbf{H} = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \tag{8.16}$$

Chacune des 3 lignes de la matrice de parité correspond à une équation de parité (addition modulo 2) liant différents bits de la séquence de sortie $\mathbf{c} = (c_1, c_2, c_3, c_4, c_5, c_6, c_7) = (u_1, u_2, u_3, u_4, c_5, c_6, c_7)$.

$$\begin{cases} u_1 + u_2 + u_3 + c_5 = 0 & \text{nœud } T_1 \\ u_2 + u_3 + u_4 + c_6 = 0 & \text{nœud } T_2 \\ u_1 + u_2 + u_4 + c_7 = 0 & \text{nœud } T_3 \end{cases} \tag{8.17}$$

On retrouve les équations de parité calculées dans l'exemple 3.

8.4 Décodage à entrées dures des codes en bloc linéaires binaires

Le décodage à entrées dures signifie que les échantillons en entrée du décodeur sont binaires. Ceci correspond au cas d'un canal à sorties binaires ou à l'application d'un seuillage des échantillons en sortie du canal de transmission.

Le décodage à entrées pondérées signifie que les échantillons en entrée du décodeur sont d'amplitude continus ou quantifiés sur quelques bits (au moins 3 bits en général)

Dans ce chapitre, nous nous intéresserons uniquement au décodage à entrées dures.

Le mot reçu $\mathbf{y}$ est la somme modulo 2 du mot de code émis $\mathbf{c}$ et d'un vecteur d'erreurs $\mathbf{e}$

$$\mathbf{y} = \mathbf{c} + \mathbf{e} \tag{8.18}$$

Une première approche pour le décodage consiste à comparer le mot reçu $\mathbf{y}$ avec l'ensemble des 2^K mots de code du code $\mathcal{C}$. Pour minimiser la probabilité d'erreurs mots, le décodeur choisit alors comme mot de code estimé $\hat{\mathbf{c}}$ le mot de code le plus près du mot reçu $\mathbf{y}$ au sens de la distance de Hamming. Cette approche est cependant très complexe à mettre en œuvre et présente un interêt pratique très limité.

En multipliant le mot reçu $\mathbf{y}$ par la matrice de parité transposée $\mathbf{H}^T$, on obtient le syndrome d'erreurs $\mathbf{s}$ de dimension $1 \times (N-K)$:

$$\begin{aligned} \mathbf{s} &= \mathbf{y}\mathbf{H}^T \\ &= \mathbf{c}\mathbf{H}^T + \mathbf{e}\mathbf{H}^T \\ &= \mathbf{e}\mathbf{H}^T \qquad \text{car} \quad \mathbf{c}\mathbf{H}^T = 0 \end{aligned} \tag{8.19}$$

En l'absence d'erreurs de transmission, le syndrome d'erreurs $\mathbf{s}$ est le vecteur nul.

Nous allons présenter deux autres méthodes de décodage : la méthode du tableau standard et le décodage par syndrome.

8.4.1 Méthode du tableau standard

Puisque le syndrome d'erreurs peut prendre 2^{N-K} valeurs différentes, une valeur de syndrome d'erreurs $\mathbf{s}$ correspond à $2^N/2^{N-K} = 2^K$ vecteurs d'erreurs possibles. La méthode du tableau standard consiste à partitionner l'espace vectoriel à N dimension en 2^K classes disjointes. Chacune des classes correspond à un mot de code.

Le tableau standard 8.5 est construit de la manière suivante :
- la première ligne contient les 2^K mots de code en commençant par le mot de code nul.
- sous le mot de code nul, on liste l'ensemble des vecteurs d'erreurs à commencer par les motifs d'erreurs de poids 1 puis le cas échéant les motifs d'erreurs de poids 2 (si $N \leq 2^{N-K}$), jusqu'à avoir rempli les 2^{N-K} cases de la première colonne.
- sous chacun des mots de code de la première ligne, on fait figurer la somme du mot de code et du vecteur d'erreurs correspondant.

$\mathbf{c}_0$	$\mathbf{c}_1$	$\mathbf{c}_2$	$\cdots$	$\mathbf{c}_{2^K-1}$
$\mathbf{c}_0 + \mathbf{e}_1$	$\mathbf{c}_1 + \mathbf{e}_1$	$\mathbf{c}_2 + \mathbf{e}_1$	$\cdots$	$\mathbf{c}_{2^K-1} + \mathbf{e}_1$
$\mathbf{c}_0 + \mathbf{e}_2$	$\mathbf{c}_1 + \mathbf{e}_2$	$\mathbf{c}_2 + \mathbf{e}_2$	$\cdots$	$\mathbf{c}_{2^K-1} + \mathbf{e}_2$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
$\mathbf{c}_0 + \mathbf{e}_{2^{N-K}-1}$	$\mathbf{c}_1 + \mathbf{e}_{2^{N-K}-1}$	$\mathbf{c}_2 + \mathbf{e}_{2^{N-K}-1}$	$\cdots$	$\mathbf{c}_{2^K-1} + \mathbf{e}_{2^{N-K}-1}$

TAB. 8.2 – Tableau standard

Chaque rangée fournit un sous ensemble de mots ou classe (*coset* en anglais) correspondant à un syndrome et un représentant commun appelé chef de classe(*coset leader* en anglais). Le chef de classe est le vecteur d'erreurs le plus vraisemblable parmi les mots de ce sous-ensemble.

Le décodage consiste donc à rechercher dans le tableau la colonne dans laquelle se trouve le mot reçu. Le résultat du décodage sera le mot de code situé en première ligne de cette colonne.

Cette méthode reste encore très complexe et a surtout un interêt pédagogique !

8.4.2 Décodage par syndrome

Nous avons vu que le syndrome d'erreurs peut prendre 2^{N-K} valeurs différentes. Le décodage par syndrome consiste tout d'abord à calculer le syndrome d'erreurs correspondant au mot reçu. Ensuite, on associe au syndrome le vecteur d'erreurs estimé correspondant $\hat{\mathbf{e}}$. Il suffit donc de stocker dans une table de correspondance les syndromes et vecteurs d'erreurs.

$\hat{\mathbf{e}}$	$\mathbf{s}$
$\hat{\mathbf{e}}_0$	$\mathbf{s}_0$
$\hat{\mathbf{e}}_1$	$\mathbf{s}_1$
$\hat{\mathbf{e}}_2$	$\mathbf{s}_2$
$\vdots$	$\vdots$
$\hat{\mathbf{e}}_{2^{N-K}-1}$	$\mathbf{s}_{2^{N-K}-1}$

TAB. 8.3 – Table de syndrome

Le mot de code estimé $\hat{\mathbf{c}}$ est alors :

$$\hat{\mathbf{c}} = \mathbf{y} + \hat{\mathbf{e}} \tag{8.20}$$

Cette méthode bien que moins complexe que la méthode du tableau standard n'est cependant applicable que pour des codes relativement simples (codes sachant corriger 1 ou 2 erreurs au maximum). Il faut en effet effectuer le produit matriciel $\mathbf{yH}^T$ et mémoriser 2^{N-K} vecteurs d'erreurs. Par exemple la table pour le décodage d'un code de Golay (23,12) nécessite une mémoire de 2048 mots de 23 bits.

exemple (suite) : considérons le code $\mathcal{C}_4$ (5,2) défini par la matrice génératrice suivante :

$$\mathbf{G} = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \end{pmatrix} \tag{8.21}$$

et la matrice de parité associée $\mathbf{H}$ suivante :

$$\mathbf{H} = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{pmatrix} \tag{8.22}$$

Le tableau standard est donné ci-dessous :

00000	01011	10101	11110	**000**
00001	01010	10100	11111	**001**
00010	01001	10111	11100	**010**
00100	01111	10001	11010	**100**
01000	00011	11101	10110	**011**
10000	11011	00101	01110	**101**
11000	10011	01101	00110	**110**
10010	11001	00111	01100	**111**

TAB. 8.4 – Tableau standard pour le code (5,2)

Il faut souligner qu'il existe plusieurs choix pour les deux dernières lignes du tableau.

Considérons le mot d'information $\mathbf{u} = [11]$.

Le mot de code associé est $\mathbf{c} = [11110]$.

On peut vérifier que le syndrome associé à ce mot de code est nul : $\mathbf{s} = [000]$

Supposons qu'une erreur survienne dans la transmission sur le 4-ième bit du mot émis : $\mathbf{e} = [00010]$

Le mot reçu est alors $\mathbf{y} = [11100]$

En utilisant le tableau on trouve directement $\hat{\mathbf{c}} = [11110]$. Le décodeur a pu corriger cette erreur.

Utilisons maintenant la méthode du syndrome. Pour construire la table de syndrome, nous calculons le syndrome correspondant à chaque vecteur d'erreur.

La table de décodage par syndrome est la suivante :

$\hat{\mathbf{e}}$	$\mathbf{s}$
00000	**000**
00001	**001**
00010	**010**
00100	**100**
01000	**011**
10000	**101**
11000	**110**
10010	**111**

TAB. 8.5 – Table de syndrome pour le code (5,2)

Le calcul du syndrome associé au mot reçu $\mathbf{y}$ donne $\mathbf{s} = \mathbf{y}\mathbf{H}^T = [010]$.

En utilisant cette table de décodage par syndrome on trouve $\hat{\mathbf{e}} = [00010]$. En ajoutant le vecteur d'erreurs estimé $\hat{\mathbf{e}}$ au mot reçu $\mathbf{r}$ on retrouve bien $\hat{\mathbf{c}} = [11110]$.

On peut voir que ce code permet de corriger tous les motifs d'erreurs simples.

8.4.3 Capacité de correction d'erreurs d'un code linéaire binaire en bloc

Le nombre d'erreurs e qu'est capable de corriger un code correcteur d'erreurs dépend de la distance minimale du code.

Les 2^K mots de code du code peuvent être vus comme les centres de boules de Hamming de rayon e. Pour garantir que les 2^K sphères ne se chevauchent pas, on doit garantir :

$$e = \left\lfloor \frac{d_{min} - 1}{2} \right\rfloor \tag{8.23}$$

La démonstration est laissée au lecteur.

Si le mot reçu est à l'intérieur de la boule de Hamming du mot de code émis alors le décodeur pourra retrouver le mot de code émis. Ceci n'est possible que si la distance entre le mot de code émis et le mot reçu est inférieure ou égale à e. En conclusion, un code en bloc linéaire binaire (N, K) de distance minimale d_{min} est capable de corriger e erreurs suivant la relation (8.23).

Par ailleurs ce même code peut aussi être utilisé pour détecter jusqu'à $d_{min} - 1$ erreurs.

exemple : les codes $\mathcal{C}_1$ (3,1) et $\mathcal{C}_3$ (7,4) dont la distance minimale est égale à 3 permettent soit de corriger une erreur soit d'en détecter deux.

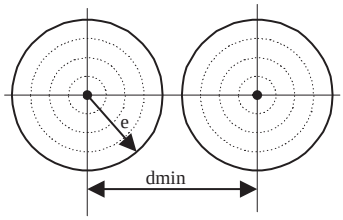


FIG. 8.2 – boules de Hamming

marge inférieure de Hamming : on a la relation suivante [1] entre K, N et e le nombre d'erreurs que peut corriger un code (N, K) :

$$2^N \geq 2^K \sum_{i=0}^{e} C_N^i \tag{8.24}$$

ou en divisant les deux termes par 2^K

$$2^{N-K} \geq \sum_{i=0}^{e} C_N^i \tag{8.25}$$

démonstration : le nombre de mots contenus dans une boule de Hamming de rayon e est égal à :

$$C_N^0 + C_N^1 + C_N^2 + \cdots + C_N^e = \sum_{i=0}^{e} C_N^i \tag{8.26}$$

Par ailleurs, le nombre total des mots contenus dans les 2^K boules de Hamming ne peut excéder 2^N pour éviter le chevauchement de ces boules. En conséquence un code corrigeant e erreurs doit satisfaire l'inégalité (8.24).

codes parfaits : les codes parfaits possèdent la propriété que tous les 2^N mots possibles sont inclus dans les 2^K boules de Hamming de rayon e. L'inégalité (8.24) se transforme en égalité.

8.5 Codes en bloc principaux et représentation graphique

8.5.1 Codes de Hamming

Les codes de Hamming sont des codes en bloc linéaires parfaits binaires (N, K) avec $N = 2^J - 1$ et $K = 2^J - 1 - J$. Un code de Hamming (N, K) peut être décrit simplement à partir de sa matrice de contrôle $\mathbf{H}$ de dimension $J \times N$. En effet, les colonnes de $\mathbf{H}$ sont les N vecteurs binaires non nuls avec J éléments.

Par exemple pour $J = 3$, le code de Hamming est un code (7,4). La matrice de parité est la suivante :

$$\mathbf{H} = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \tag{8.27}$$

La distance de Hamming de ces codes est égale à 3 et ils peuvent donc corriger une erreur. Au décodage, une valeur non nulle du syndrome donne directement la position de l'erreur. En effet, le tableau standard contient 8 lignes (une ligne pour le cas sans erreur plus 7 lignes correspondant chacune à une position de l'erreur).

Nous verrons que les codes de Hamming appartiennent à la classe des codes cycliques et peuvent aussi être construits à partir d'un polynôme générateur.

[1] $C_n^p = \frac{n!}{p!(n-p)!}$ est le nombre de combinaisons sans répétition de n éléments pris p à p

8.5.2 Code de Golay

Le code de Golay est un code linéaire binaire (23,12) dont la distance minimale est égale à 7. Ce code est le seul code parfait binaire avec les codes de Hamming. En effet, pour $N = 23$, $K = 12$ et $e = 3$ on a l'égalité :

$$1 + C_{23}^{1} + C_{23}^{2} + C_{23}^{3} = 2^{11}$$

Les codes de Golay sont aussi des codes cycliques (voir chapitre suivant).

La matrice génératrice systématique du code de Golay (23,12) est la suivante :

$$\mathbf{G} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \tag{8.28}$$

Sa fonction d'énumération WEF est la suivante :

$$A(D) = 1 + 253D^{7} + 506D^{8} + 1288D^{11} + 1288D^{12} + 506D^{15} + 253D^{16} + D^{23} \tag{8.29}$$

A partir du code de Golay (23,12), il est possible de construire le code de Golay étendu (24,12) en ajoutant un bit de parité. La distance minimale de ce code est égale à 8. La matrice génératrice non systématique du code de Golay étendue (24,12) est la suivante :

$$\mathbf{G} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \tag{8.30}$$

Sa fonction d'énumération WEF est la suivante :

$$A(D) = 1 + 759D^{8} + 2576D^{12} + 759D^{16} + D^{24} \tag{8.31}$$

8.5.3 Représentations graphiques des codes en bloc linéaires binaires

D'une manière générale, tout code linéaire binaire peut être représenté sous la forme d'un graphe de Tanner. Un graphe de Tanner est un graphe biparti comprenant 2 types de nœuds : des nœuds de variable binaire représentés par un cercle et des nœuds de contrôle de parité représentés

par un carré avec un "+" à l'intérieur. Chaque branche signifie une dépendance entre le nœud de variable et le nœud de contrôle qu'elle relie.

Le graphe de Tanner se déduit directement de la matrice de parité $\mathbf{H}$: chaque nœud de contrôle de parité i du graphe de Tanner correspond à la i-ième ligne de la matrice de parité $\mathbf{H}$ et chaque nœud de variable j correspond à la j-ième colonne de $\mathbf{H}$. En effet, chaque ligne de $\mathbf{H}$ définit une équation de parité entre plusieurs variables. Une branche reliera le nœud de contrôle de parité i avec le nœud de variable j si et seulement si $h_{ij} = 1$.

Comme il y a plusieurs matrices de parité $\mathbf{H}$ pour un même code, il y aura autant de graphes de Tanner associés.

Exemple : considérons le code de Hamming $(7, 4)$ défini par le polynôme générateur $g(D) = 1 + D^2 + D^3$ Comme nous l'avons vu précédemment, la matrice génératrice de ce code sous sa forme systématique est la suivante :

$$\mathbf{G} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} \tag{8.32}$$

et la matrice de parité associée $\mathbf{H}$

$$\mathbf{H} = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \tag{8.33}$$

Chacune des 3 lignes de la matrice de parité correspond à une équation de parité (addition modulo 2) liant différents bits de la séquence de sortie $\mathbf{c} = (c_1, c_2, c_3, c_4, c_5, c_6, c_7) = (u_1, u_2, u_3, u_4, c_5, c_6, c_7)$.

$$\begin{cases} u_1 + u_2 + u_3 + c_5 = 0 & \text{nœud } T_1 \\ u_2 + u_3 + u_4 + c_6 = 0 & \text{nœud } T_2 \\ u_1 + u_2 + u_4 + c_7 = 0 & \text{nœud } T_3 \end{cases} \tag{8.34}$$

Le graphe de Tanner associé à ces équations de parité est donné sur la figure 8.3.

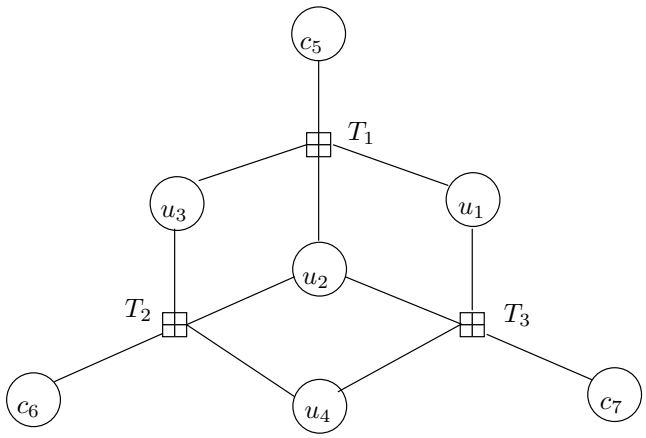


FIG. 8.3 – Graphe de Tanner du code de Hamming (7,4)

Ces représentations graphiques s'avèrent être des outils très pratiques pour étudier les codes correcteurs d'erreurs et analyser les algorithmes de décodage.

Pour terminer ce chapitre, nous allons montrer comment à partir de la matrice génératrice d'un code linéaire en bloc il est possible de représenter ce code par un diagramme en treillis. Pour ce

0000000	0001011	0010110	0011101
0100111	0101100	0110001	0111010
1000101	1001110	1010011	1011000
1100010	1101001	1110100	1111111

TAB. 8.6 – Liste des mots de code du code de Hamming (7,4)

faire, il faut tout d'abord mettre la matrice génératrice sous une forme adaptée à la représentation en treillis.

Soit l'enveloppe d'un mot de code la séquence de bits à 1 qui commence au premier bit non nul du mot de code et se termine au dernier bit non nul du mot de code.

Par exemple l'enveloppe associée au mot de code 1101000 est 1111000 .

Une matrice génératrice sera dite orientée treillis si elle permet de construire un diagramme en treillis avec le minimum de branche.

exemple : considérons le code de Hamming (7,4) défini par le polynôme générateur $g(D) = 1 + D + D^3$

$$\mathbf{G} = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix} \tag{8.35}$$

Sa matrice génératrice sous la forme systématique est la suivante

$$\mathbf{G_s} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix} \tag{8.36}$$

La matrice enveloppe associée à la matrice génératrice systématique $\mathbf{G_s}$ est

$$\mathbf{E_S} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \tag{8.37}$$

Pour déterminer le nombre de branches du treillis, on commence par compter le nombre de 1 par colonne de la matrice enveloppe. Soit n_i le nombre de 1 de la i-ème colonne ; le nombre total de branches est égal à $\sum_i 2^{n_i}$.

Pour cet exemple, le nombre de branches du treillis est égal à $2^1+2^2+2^3+2^4+2^4+2^4+2^3 = 70$.

La matrice enveloppe associée à la matrice génératrice $\mathbf{G}$ est la suivante

$$\mathbf{E} = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \tag{8.38}$$

Le nombre de branche du treillis correspondant est égal à $2^1 + 2^2 + 2^3 + 2^4 + 2^3 + 2^2 + 2^1 = 44$.

Comme la complexité du décodage à partir du trellis du code est proportionnelle au nombre de branche de celui-ci, il est préférable d'utiliser la seconde matrice génératrice.

Pour construire le treillis de ce code, il faut considérer chaque ligne de la matrice génératrice comme un sous-code (N,1) et ne comprenant donc que 2 mots de code.

Construisons le treillis à partir de la matrice génératrice **G**. Les différentes phases de cette construction sont présentées sur la figure 8.4. Pour la première ligne, le sous-code comprend les mots de code 0000000 et 11010000. Le treillis de ce sous-code comprend donc 2 chemins. On construit alors le treillis du sous-code correspondant aux deux premières lignes de matrice génératrice. Ce treillis est simplement le produit du treillis des sous-codes relatifs à la première et seconde ligne du treillis. Cette procédure est répétée jusqu'à construction complète du treillis. On peut vérifier que le treillis ainsi obtenu comporte bien 44 branches. Nous verrons dans le paragraphe consacré au décodage des codes convolutifs qu'il existe des algorithmes de décodage utilisant la représentation en treillis. Ces algorithmes peuvent donc aussi être utilisés pour le décodage des codes en bloc linéaires.

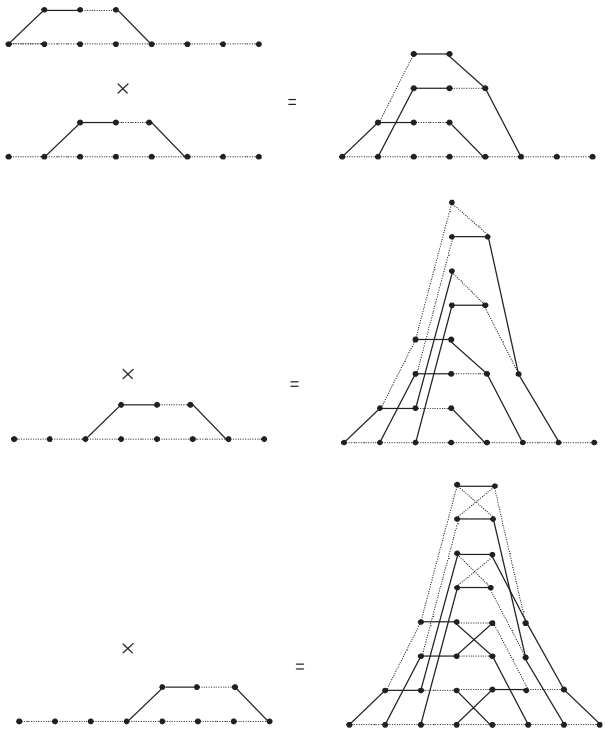


FIG. 8.4 – Diagramme en treillis du code de Hamming (7,4)

8.6 Bornes sur la distance minimale des codes linéaires en bloc

Nous allons donner dans ce paragraphe les bornes inférieures et supérieures principales sur la distance minimale des codes linéaires en bloc. Ces bornes sont très utiles pour comparer les codes correcteurs d'erreurs entre eux.

Une première borne supérieure est la borne de Singleton :

$$d_{min} \leq N - K + 1 \tag{8.39}$$

Les codes réalisant l'égalité dans (8.39) sont appelés codes séparables à distance maximale ou code MDS.

En divisant les deux termes par N, on obtient :

$$\frac{d_{min}}{N} \leq 1 - R + \frac{1}{N} \quad \text{avec} \quad R = \frac{K}{N} \tag{8.40}$$

Une seconde borne supérieure est la borne de Plotkin qui s'obtient lorsque $N \to +\infty$:

$$\frac{d_{min}}{N} \leq \frac{1}{2}(1 - R) \tag{8.41}$$

Une borne supérieure plus fine est la borne de McEliece-Rodemich-Rumsey-Welch [19] :

$$R \leq 1 - H_2\left(\frac{1}{2} - \sqrt{\frac{d_{min}}{N}\left(1 - \frac{d_{min}}{N}\right)}\right) \qquad \forall N \tag{8.42}$$

avec $H_2(\alpha) = -\log_2 \alpha - (1-\alpha)\log_2(1-\alpha)$

Finalement nous présentons la borne de Gilbert-Varshamov qui est la borne inférieure la plus utilisée :

$$R \geq 1 - H_2\left(\frac{d_{min}}{N}\right) \qquad \forall N \tag{8.43}$$

Sur la figure 8.6 nous présentons les courbes $R = f(d_{min}/N)$ relatives aux bornes de Gilbert-Varshamov et de McEliece-Rodemich-Rumsey-Welch.

Rappelons que les bornes de Plotkin, Gilbert-Varshamov et McEliece-Rodemich-Rumsey-Welch sont valables lorsque $N \to +\infty$.

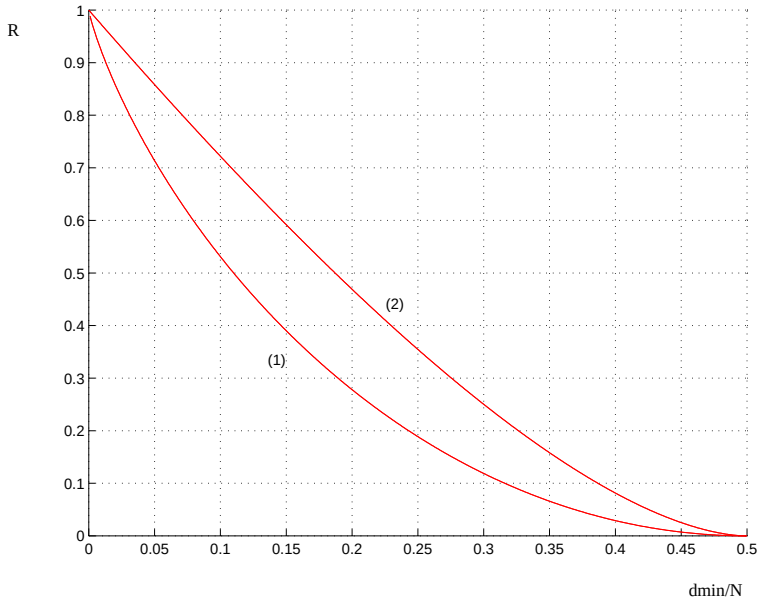


FIG. 8.5 – bornes sur la distance minimale des codes linéaires en bloc (1) borne inférieure de Gilbert-Varshamov (2) borne supérieure de McEliece

8.7 Décodage optimal

L'objectif du décodage optimal est de déterminer la séquence qui a été le plus vraisemblablement émise $\hat{\mathbf{x}}$.

Soit la séquence $\mathbf{x} = (x_0, x_1, x_2, \ldots, x_{N-1})$ de longueur N envoyée dans un canal discret stationnaire sans mémoire de densité de probabilité conditionnelle $p(y/x)$ et $\mathbf{y} = (y_0, y_1, y_2, \ldots, y_{N-1})$ la séquence reçue.

D'une manière générale, un décodeur *maximum a posteriori* (MAP) cherche parmi toutes les séquences possibles $\mathbf{x}$, la séquence estimée $\hat{\mathbf{x}}$ pour laquelle la probabilité conditionnelle $Pr(\mathbf{x}|\mathbf{y})$ est la plus grande.

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} Pr(\mathbf{x}|\mathbf{y}) \tag{8.44}$$

On peut écrire :

$$Pr(\mathbf{x}|\mathbf{y}) = \frac{Pr(\mathbf{y}|\mathbf{x})Pr(\mathbf{x})}{Pr(\mathbf{y})} \tag{8.45}$$

Il est raisonnable de considérer que tous les mots de code sont a priori équiprobables.

Avec cette hypothèse et puisque de plus le dénominateur $Pr(\mathbf{y})$ est commun à toutes les séquences, la séquence estimée $\hat{\mathbf{x}}$ est la séquence pour laquelle la probabilité conditionnelle $Pr(\mathbf{y}|\mathbf{x})$ est la plus grande.

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} Pr(\mathbf{y}|\mathbf{x}) \tag{8.46}$$

Un décodeur utilisant ce critère est appelé un décodeur à *maximum de vraisemblance* (*maximum likelihood* en anglais ou ML).

Lorsque les mots de code sont équiprobables, les décodeurs MAP et ML sont identiques.

En considérant que les signaux émis sont perturbés indépendamment les uns des autres, la probabilité conditionnelle est égale au produit des probabilités conditionnelles $Pr(y_i|x_i)$:

$$Pr(\mathbf{y}|\mathbf{x}) = \prod_{i=0}^{N-1} Pr(y_i|x_i) \tag{8.47}$$

La recherche de la séquence la plus probable implique donc que le décodeur ML calcule les distances entre la séquence reçue et toutes les séquences possibles. Il faut ici distinguer deux cas :
- si l'entrée du décodeur est binaire on parlera de décodage à entrées dures (*hard decoding* en anglais). Ce cas correspond à l'utilisation d'un canal à sorties binaires (comme le canal binaire symétrique par exemple) ou d'un canal à sorties continues mais ayant subi un seuillage avant le décodeur. Le décodeur à entrées dures calcule des distances de Hamming
- si l'entrée du décodeur peut prendre une valeur continue, on dira que le décodage est à entrées souples ou pondérées (*soft decoding* en anglais). Ce cas correspond par exemple à l'utilisation d'un canal à sorties continues comme le canal à bruit blanc additif gaussien. En pratique, l'entrée est quantifiée sur plusieurs bits (3 ou 4 bits suffisent généralement pour ne pas dégrader les performances du décodeur. Contrairement au décodeur à entrées dures, le décodeur à entrées souples calcule des distances euclidiennes.

8.8 Performances des codes linéaires en bloc avec décodage à entrées dures

Pour le canal binaire symétrique de probabilité d'inversion p, la probabilité $Pr(\mathbf{y}|\mathbf{x})$ est la suivante :

$$Pr(\mathbf{y}|\mathbf{x}) = p^{d_H(\mathbf{y},\mathbf{x})}(1-p)^{N-d_H(\mathbf{y},\mathbf{x})} = (1-p)^N \left(\frac{p}{1-p}\right)^{d_H(\mathbf{y},\mathbf{x})} \tag{8.48}$$

où $d_H(\mathbf{y}, \mathbf{x})$ est la distance de Hamming entre la séquence reçue $\mathbf{y}$ et la séquence $\mathbf{x}$. Comme p est compris entre 0 et 0.5, on a $0 < \frac{p}{1-p} < 1$. Ainsi, maximiser $Pr(\mathbf{y}|\mathbf{x})$ revient à minimiser la distance de Hamming entre $\mathbf{y}$ et $\mathbf{x}$.

Pour un canal de transmission binaire symétrique, la probabilité qu'un mot reçu de longueur N bits contienne i erreurs est la suivante :

$$P_{Ni} = C_N^i p^i (1-p)^{N-i} \tag{8.49}$$

où p est la probabilité d'erreurs du canal binaire symétrique.

Ainsi donc, pour un code en bloc linéaire sachant corriger jusqu'à e erreurs dans le mot reçu de longueur N on peut donner une borne supérieure sur la probabilité d'erreurs mots après décodage TEM en sommant les probabilités P_{Ni} correspondant aux cas où le nombre d'erreurs en sortie du canal est supérieur à la capacité de correction du code. On a :

$$TEM \leq \sum_{i=e+1}^{N} P_{Ni} = \sum_{i=e+1}^{N} C_N^i p^i (1-p)^{N-i} \tag{8.50}$$

L'inégalité se transforme en une égalité uniquement lorsqu'un code parfait est utilisé.

8.9 Bornes par réunion

La borne par réunion permet de majorer une probabilité d'erreurs.

Considérons un décodeur à maximum de vraisemblance qui décide en faveur du mot de code le plus probable c'est-à-dire le plus proche au sens de la distance euclidienne :

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} Pr(\mathbf{y}|\mathbf{x}) \tag{8.51}$$

La probabilité de décoder un mauvais mot de code sachant un mot de code $\mathbf{x}_i$ transmis est bornée supérieurement comme suit :

$$\begin{aligned} Pr(\text{erreur mot}|\mathbf{x}_i) &= Pr(\mathbf{y} \in \bar{\Lambda}_i|\mathbf{x}_i) \\ &\leq \sum_{j:j\neq i} Pr(\mathbf{y} \in \Lambda_j|\mathbf{x}_i) \end{aligned} \tag{8.52}$$

avec Λ_i zone de décision associée au mot de code $\mathbf{x}_i$.

la probabilité $Pr(\mathbf{y} \in \Lambda_j|\mathbf{x}_i)$ que $\mathbf{y}$ soit plus près de $\mathbf{x}_j$ que de $\mathbf{x}_i$ est appelée la probabilité d'erreurs par paire. On la note $Pr(\mathbf{x}_i \to \mathbf{x}_j)$.

$$Pr(\mathbf{x}_i \to \mathbf{x}_j) \stackrel{\text{def}}{=} Pr(\mathbf{y} \in \Lambda_j|\mathbf{x}_i) \tag{8.53}$$

Sachant que $\mathbf{x}_i$ a été transmis, la probabilité que $\mathbf{y}$ ne soit pas dans la zone de décision Λ_i est inférieure ou égale à la somme des probabilités d'erreurs par paire $Pr(\mathbf{x}_i \to \mathbf{x}_j)$ pour tout $j \neq i$.

Finalement le taux d'erreurs mot (TEM) est égal à :

$$\begin{aligned} TEM &= \sum_i Pr(\mathbf{x}_i, \text{erreur mot}) \\ &= \sum_i Pr(\mathbf{x}_i) Pr(\text{erreur mot}|\mathbf{x}_i) \end{aligned} \tag{8.54}$$

D'une manière générale, la borne par réunion est un outil très utile pour évaluer les performances des systèmes de transmission avec codage même si elle n'est précise que pour les rapports signal à bruit élevés.

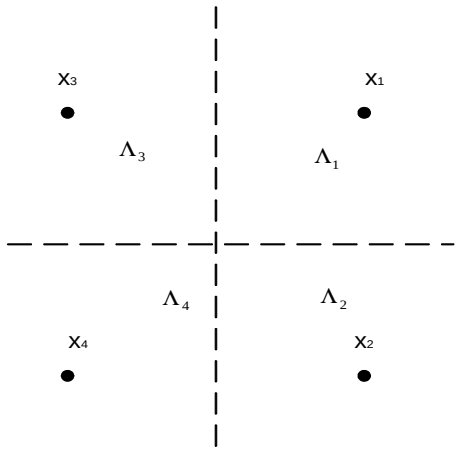


FIG. 8.6 – Zones de décision associées aux mots de code.

Exemple : soit un ensemble de 4 mots de codes $\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3}$ et $\mathbf{x_4}$ et leurs zones de décision associées $\Lambda_1, \Lambda_2, \Lambda_3$ et Λ_4 présenté sur la figure 8.6 .

On peut par exemple borner supérieurement la probabilité d'erreur $Pr(\text{erreur mot}|\mathbf{x}_1)$:

$$\begin{aligned} Pr(\text{erreur mot}|\mathbf{x}_1) &= Pr(\mathbf{y} \in \bar{\Lambda_1}|\mathbf{x}_1) \\ &= Pr(\mathbf{y} \in (\Lambda_2 \cup \Lambda_3 \cup \Lambda_4)|\mathbf{x}_1) \\ &\leq Pr(\mathbf{y} \in \Lambda_{12}|\mathbf{x}_1) + Pr(\mathbf{y} \in \Lambda_{13}|\mathbf{x}_1) + Pr(\mathbf{y} \in \Lambda_{14}|\mathbf{x}_1) \\ &= Pr(\mathbf{x}_1 \to \mathbf{x}_2) + Pr(\mathbf{x}_1 \to \mathbf{x}_3) + Pr(\mathbf{x}_1 \to \mathbf{x}_4) \end{aligned}$$

L'exemple est illustré par la figure 8.7 .

8.10 Performances des codes linéaires en bloc avec décodage à entrées souples

Considérons à nouveau une modulation bipodale où les échantillons codés émis peuvent prendre les amplitudes $+\sqrt{RE_b}$ ou $-\sqrt{RE_b}$ (avec E_b énergie de l'échantillon et $R = \frac{K}{N}$ est le rendement du code.) sur un canal BBAG. A la sortie du filtre adapté, on a :

$$y_i = x_i + n_i$$

On rappelle que pour un canal BBAG la probabilité $Pr(\mathbf{x}_i \to \mathbf{x}_j)$ est donnée par :

$$Pr(\mathbf{x}_i \to \mathbf{x}_j) = \frac{1}{2}\text{erfc}\left(\frac{d(\mathbf{x}_i, \mathbf{x}_j)}{2\sqrt{N_0}}\right) \tag{8.55}$$

où $d(\mathbf{x}_i, \mathbf{x}_j)$ est la distance euclidienne entre les deux mots $\mathbf{x}_i$ et $\mathbf{x}_j$.

Soit $\mathbf{c}_i$ et $\mathbf{c}_j$ les deux mots de code associés respectivement aux mots $\mathbf{x}_i$ et $\mathbf{x}_j$. Si la distance de Hamming entre $\mathbf{c}_i$ et $\mathbf{c}_j$ est égale à $d_H(\mathbf{c}_i,\mathbf{c}_j) = d$, la distance euclidienne entre $\mathbf{x}_i$ et $\mathbf{x}_j$ est égale à

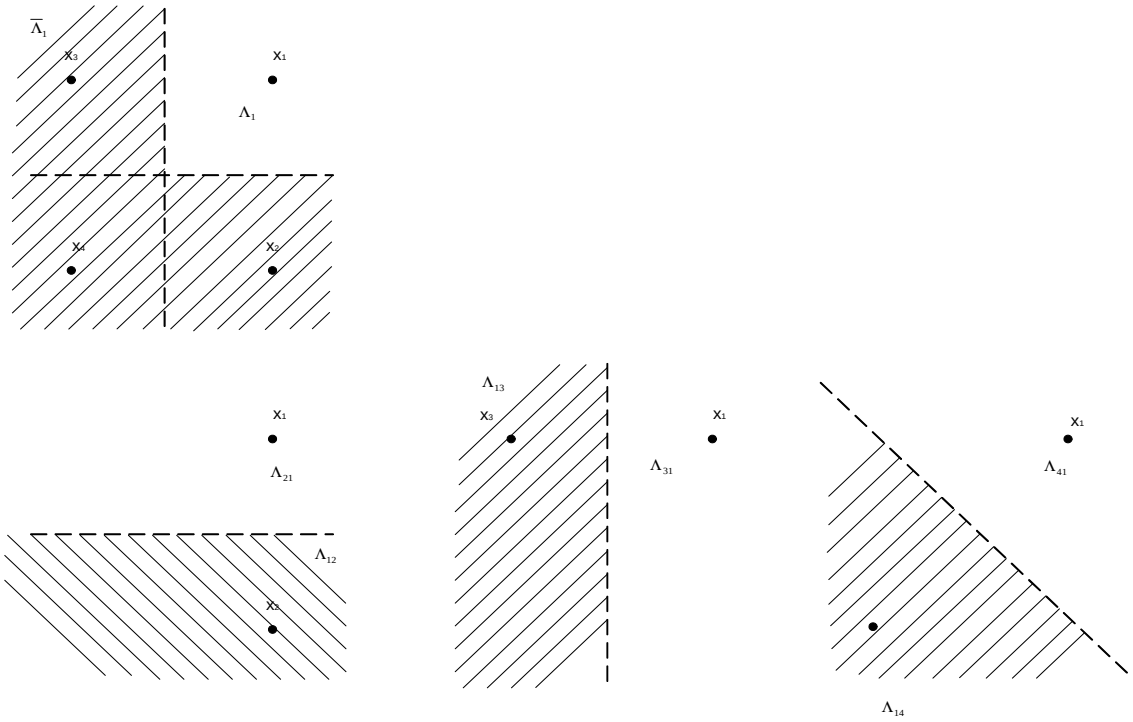


FIG. 8.7 – Exemple.

$$\begin{aligned} d(\mathbf{x}_i, \mathbf{x}_j) &= \sqrt{\sum_{k=1}^{d} (2\sqrt{RE_b})^2} \\ &= \sqrt{4dRE_b} \\ &= 2\sqrt{dRE_b} \end{aligned} \tag{8.56}$$

On a alors :

$$Pr(\mathbf{x}_i \to \mathbf{x}_j) = \frac{1}{2}\text{erfc}\left(\sqrt{dR\frac{E_b}{N_0}}\right) \tag{8.57}$$

Exemple1 : code de répétition (3,1)
TBD

Exemple2 : code de parité (3,2)
TBD

Nous allons maintenant utiliser la borne par réunion pour exprimer une borne supérieure sur le taux d'erreur mot (TEM) et le taux d'erreur bit (TEB) du décodeur à maximum de vraisemblance sur un canal à bruit blanc additif (BBAG) associé à un code en bloc linéaire (N, K). La borne par réunion ramène la comparaison d'un ensemble de mots de code à un certain nombre de comparaison de mots de code deux à deux. En supposant que tous les mots de code sont équiprobables, on obtient la borne supérieure suivante sur le taux d'erreur mot :

$$TEM \leq \frac{1}{2} \sum_{d=d_{min}}^{N} A_d \text{erfc}\left(\sqrt{dR\frac{E_b}{N_0}}\right) \tag{8.58}$$

où A_d est le nombre de mots de code de poids d. d_{min} est la distance de Hamming libre ou minimale du code en bloc.

Par ailleurs il est possible d'exprimer une première borne supérieure de type réunion-Bhattacharyya [30] sur le taux d'erreur bit :

$$TEB \leq \frac{W}{K}\frac{\partial A^C(W,Z)}{\partial W}\bigg|_{W=Z=e^{-RE_b/N_0}} \tag{8.59}$$

$$= \sum_{d=d_{min}}^{N} B_d H^d \bigg|_{H=e^{-RE_b/N_0}} \tag{8.60}$$

avec

$$B_d = \sum_{d:d=w+z} \frac{w}{K} A_{w,z} \tag{8.61}$$

La première ligne permet de distinguer la contribution des mots d'information de poids w alors que dans la seconde ligne nous avons regroupé la contribution des mots de code de poids d en introduisant le coefficient B_d.

Une borne plus fine sur le taux d'erreur bit [30] peut être obtenue à partir de (8.10) en utilisant l'inégalité suivante :

$$\text{erfc}(\sqrt{x+y}) \leq \text{erfc}(\sqrt{x})e^{-y}$$

On a alors :

$$\text{erfc}\left(\sqrt{dR\frac{E_b}{N_0}}\right) \leq \text{erfc}\left(\sqrt{d_{min}R\frac{E_b}{N_0}}\right)e^{-\frac{(d-d_{min})RE_b}{N_0}}$$

Finalement, cette borne supérieure sur le taux d'erreur bit s'exprime comme suit :

$$TEB \leq \frac{W}{2K}\text{erfc}\left(\sqrt{d_{min}R\frac{E_b}{N_0}}\right)e^{d_{min}R\frac{E_b}{N_0}}\frac{\partial A^C(W,Z)}{\partial W}\bigg|_{W=Z=e^{-R\frac{E_b}{N_0}}}$$

$$= \frac{1}{2}\text{erfc}\left(\sqrt{d_{min}R\frac{E_b}{N_0}}\right)e^{d_{min}R\frac{E_b}{N_0}}\sum_{d=d_{min}}^{N} B_d H^d\bigg|_{H=e^{-R\frac{E_b}{N_0}}} \tag{8.62}$$

En ne gardant que les premiers termes on utilisera l'approximation suivante :

$$\boxed{TEB \approx \frac{1}{2}\sum_{d=d_{min}}^{N} B_d \text{erfc}\left(\sqrt{dR\frac{E_b}{N_0}}\right)} \tag{8.63}$$

Traditionnellement on considère qu'un bon code correcteur d'erreur est un code qui maximise la distance minimale d_{min}. Cette distance conditionne les performances à fort rapport E_b/N_0. Une autre approche permettant aussi de diminuer le TEB consiste à réduire B_d.

exemple (suite) : code de parité (3,2)

A partir de la fonction d'énumération de poids $A(W,Z)$ on obtient

$$B_2 = \frac{1}{2}2 + \frac{2}{2}1 = 2$$

$$TEM \leq \frac{3}{2}\text{erfc}\left(\sqrt{\frac{4}{3}\frac{E_b}{N_0}}\right) \tag{8.64}$$

et

$$TEB \approx \text{erfc}\left(\sqrt{\frac{4}{3}\frac{E_b}{N_0}}\right) \tag{8.65}$$

Remarque : certains auteurs distinguent l'énergie par bit transmis E_b et l'énergie par bit utile ou bit d'information E_{but}, dans ce document nous n'utiliserons pas la notation E_{but} et donc E_b sera toujours **l'énergie par bit utile** ou d'information.

8.11 Gain de codage

Nous avons vu que pour un canal BBAG, il est théoriquement possible de réaliser une transmission sans erreur à un rapport $E_b/N_0 = 0dB$ en utilisant un code de rendement $1/2$. La différence entre une chaîne de transmission utilisant une modulation bipodale sans codage et cette limite de capacité théorique est de 9.6 dB (à un taux d'erreur bit de 10^{-5}). L'ajout d'un code correcteur d'erreur doit permettre de rapprocher le point de fonctionnement de la chaîne de transmission de cette limite théorique.

Définition : pour un taux d'erreur fixé (mot, bit, symbole), on définit le gain de codage d'un code correcteur d'erreur comme étant la différence de rapport E_b/N_0 entre la chaîne sans codage et la chaîne utilisant ce code [2] .

Sur la figure 8.8 nous avons présenté les courbes de performances théoriques $TEM = f(E_b/N_0)$ (obtenue en utilisant l'équation (8.47)) de trois chaînes de transmission utilisant un code correcteur d'erreur et un décodeur à entrées pondérées ainsi que la courbe obtenue sans codage (modulation bipodale). Pour un taux d'erreur mot de 10^{-5}, les gains de codage du code de parité (3,2), du code de Hamming (7,4) et du code de Golay (23,12) sont respectivement de 1.3dB, 2.3dB et 3.8dB.

Dans le paragraphe précédent, nous avons déterminer une borne supérieure du taux d'erreurs mots :

$$TEM \leq \frac{1}{2}\sum_{d=d_{min}}^{N} A_d \text{erfc}\left(\sqrt{dR\frac{E_b}{N_0}}\right) \tag{8.66}$$

Une approximation raisonnable pour les taux d'erreurs élevés consiste à ne prendre en compte que la contribution des mots de code dont le poids est égal à la distance minimale. On a alors l'approximation suivante :

$$TEM \approx \frac{1}{2}A_{dmin}\text{erfc}\left(\sqrt{dminR\frac{E_b}{N_0}}\right) \tag{8.67}$$

Si on ne tient pas compte du terme A_{dmin}, le gain de codage "asymptopique" s'écrit simplement :

$$\text{GC}_{asympt} \approx 10\log_{10}(d_{min}R) \qquad \text{en dB} \tag{8.68}$$

En prenant en compte le nombre de mots de codes à la distance minimale A_{dmin}, on peut approximer le gain de codage réel comme suit :

$$GC_{TEM} \approx 10\log_{10}(d_{min}R) - 0.2\log_2(A_{dmin}) \qquad \text{en dB} \tag{8.69}$$

[2] Certains auteurs distinguent l'énergie par bit transmis E_b et l'énergie par bit utile ou bit d'information E_{but}, dans ce document nous n'utiliserons pas la notation E_{but} et donc E_b sera toujours **l'énergie par bit utile** ou d'information.

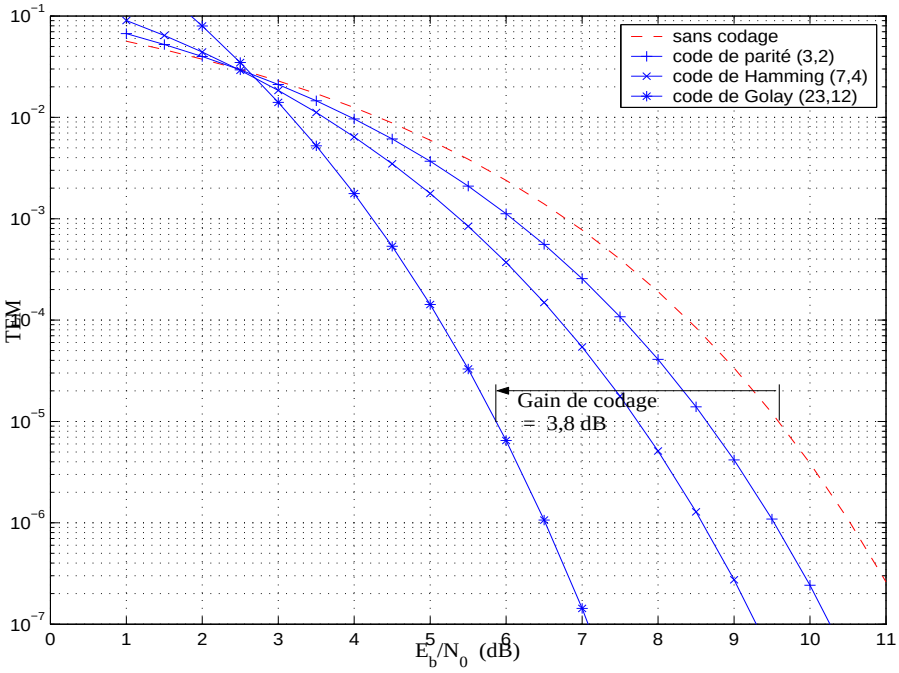


FIG. 8.8 – Gain de codage de différents codes

Le facteur $0.2\log_2(A_{dmin})$ prend en compte l'impact du nombre de mots de codes à la distance minimale sur le gain de codage. Le facteur 0.2 correspond à la pente de la fonction $erfc(.)$ dans la région $10^{-5} - 10^{-5}$. Cette formule n'est qu'une approximation.

Par exemple, pour le code de Golay (23,12), on a $R = \frac{12}{23}$, $d_{min} = 7$ et $A_{dmin} = 253$ soit un gain de codage approximativement égal à 4 dB.

Au lieu du TEM, il est également possible d'utiliser le taux d'erreurs mots par bit d'information (TEMB) pour définir le gain de codage :

$$TEMB = \frac{TEM}{K} \approx \frac{1}{2}\frac{A_{dmin}}{K}\text{erfc}\left(\sqrt{dminR\frac{E_b}{N_0}}\right) \tag{8.70}$$

Dans ce cas on a la relation suivante :

$$GC_{TEMB} \approx 10\log_{10}(d_{min}R) - 0.2\log_2(\frac{A_{dmin}}{K}) \qquad \text{en dB} \tag{8.71}$$

Pour l'exemple considéré ci-dessus, le gain de codage serait alors égal à 4,75 dB.

Sur la table suivante nous présentons les gains de codage pour différents codes correcteurs d'erreurs en bloc binaires :
Une autre approche consiste à utiliser le TEB (mais qui implique alors de calculer B_{dmin}).

8.12 Comparaison des performances des décodeurs à entrées dures et pondérées

Sur la figure 8.9 nous présentons les courbes de performances $TEM = f(E_b/N_0)$ d'une chaîne de transmission utilisant un code de Hamming (7,4) et un code de Golay (23,12). Les courbes en trait pointillé sont obtenues avec un décodeur à entrées dures (en utilisant l'équation (8.50))

code	(N, K)	R	d_{min}	A_{dmin}	GC_{asympt}	GC_{TEMB}
Hamming	$(7, 4)$	0.57	3	7	1.1 dB	0.84 dB
Reed-Muller	$(8, 4)$	0.5	4	14	3	2,6
Hamming	$(15, 11)$	0.733	3			
Reed-Muller	$(16, 5)$	0.31	8	30	4	3,5
Golay	$(23, 12)$	0.52	7	253	5,6	4,7
Golay	$(24, 12)$	0.5	8	759	6	4,8

et les courbes en trait continu sont obtenues avec un décodeur à entrées pondérées (en utilisant l'équation (8.47)). Pour le code de Golay, on peut observer que le décodage à entrées pondérées apporte un gain d'environ 2 dB.

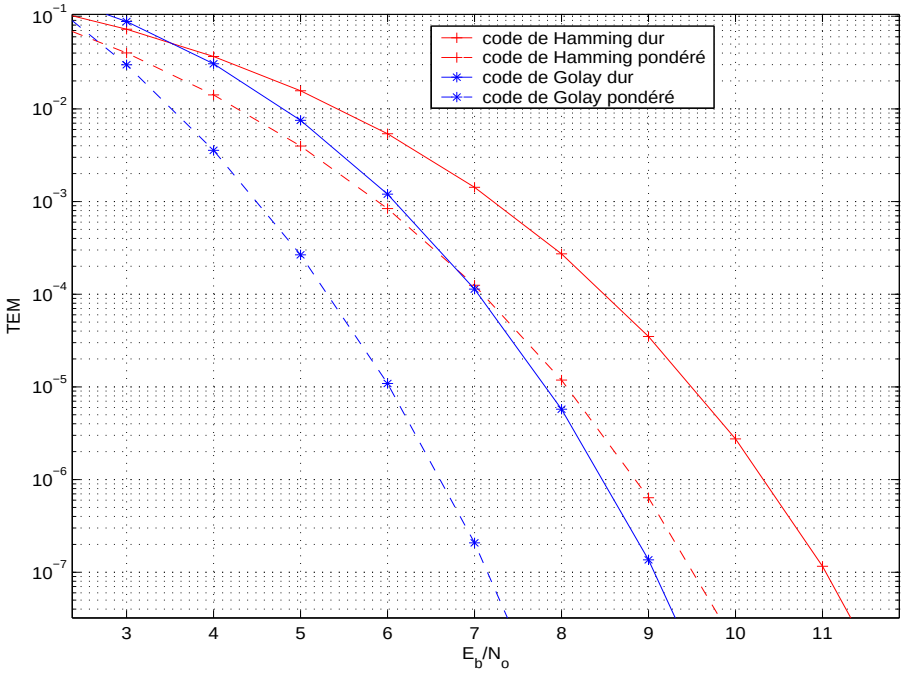


FIG. 8.9 – Comparaison du décodage à entrées dures et pondérées

Chapitre 9

Codes cycliques

9.1 Definition et Propriétés

Avertissement : Dans ce chapitre, la convention de notation adoptée est la convention poids fort (MSB)à droite.

Les codes cycliques forment un sous ensemble des codes linéaires en bloc. Alors que pour un code en bloc linéaire K mots de code sont nécessaires pour déterminer l'ensemble des 2^K mots de code, dans le cas des codes cycliques, un seul mot de code suffit.

Ces codes contiennent les familles de codes les plus importantes comme les codes de Hamming, de Golay, les codes BCH et Reed Solomon. Leurs propriétés permettent un codage et un décodage relativement aisés.

Dans ce chapitre, nous présenterons les codes cycliques dans le corps de Galois GF(2) bien que ces codes se généralisent dans GF(q).

Les codes cycliques possèdent la propriété principale suivante :

Si $\mathbf{c} = [c_0\ c_1\ \ldots\ c_{N-2}\ c_{N-1}]$ est un mot de code, alors le mot obtenu par décalage circulaire à droite d'une position $\mathbf{c'} = [c_{N-1}\ c_0\ \ldots\ c_{N-3}\ c_{N-2}]$ est aussi un mot de code.

Pour décrire un code cyclique (N, K), il est pratique d'associer à chaque mot de code $\mathbf{c} = [c_0\ c_1\ \ldots\ c_{N-2}\ c_{N-1}]$ un polynôme $c(p)$ de degré inférieur ou égal à $N-1$:

$$c(p) = c_0 + c_1 p + \cdots + c_{N-2}p^{N-2} + c_{N-1}p^{N-1}$$

Nous allons montrer que les propriétés des codes cycliques s'obtiennent simplement en utilisant l'algèbre des polynômes modulo $p^{N-1} - 1$.

Calculons le polynôme $pc(p)$:

$$pc(p) = c_0 p + c_1 p^2 + \cdots + c_{N-2}p^{N-1} + c_{N-1}p^N$$

Aucun mot de code n'est associé à ce polynôme puisque le degré de celui-ci peut être supérieur à $N-1$.

En ajoutant c_{N-1} puis retranchant c_{N-1}, cette expression peut encore s'écrire :

$$pc(p) = c_{N-1} + c_0 p + c_1 p^2 + \cdots + c_{N-2}p^{N-1} + c_{N-1}(p^N - 1)$$

Or le polynôme $c'(p)$ associé au mot de code **c'** défini ci-dessus est égal à :

$$c'(p) = c_{N-1} + c_0 p + \cdots + c_{N-2}p^{N-1}$$

Ainsi, on a donc :

$$c'(p) = pc(p) - c_{N-1}(p^N - 1)$$

Effectuons le calcul de $pc(p)$ modulo $p^N - 1$. On obtient :

$$c'(p) = pc(p) \qquad \text{mod } (p^N - 1)$$

Ainsi un décalage circulaire à droite d'une position correspond à une multiplication par p modulo $p^N - 1$.

Plus généralement un décalage circulaire à droite de i positions correspond à une multiplication par p^i modulo $p^N - 1$.

9.1.1 Propriétés des codes cycliques

Propriété 1 : Si $c(p)$ est un polynôme de degré $N-1$ associé à un mot de code d'un code cyclique (N, K), alors

$$(a_0 + a_1 p + a_2 p^2 + \cdots + a_{K-1} p^{K-1}) c(p) \qquad \text{mod } (p^N - 1) \qquad (9.1)$$
$$\text{avec} \quad a_0, a_1, \ldots, a_{K-1} \in GF(2)$$

est aussi un polynôme associé à un mot de code.

Cette relation montre qu'à partir d'un polynôme $c(p)$, il est possible de retrouver l'ensemble des 2^K mots de code du code cyclique.

Propriété 2 : Il est possible de construire un code cyclique (N, K) à partir d'un polynôme générateur noté $g(p)$ de degré $N - K$.

$$g(p) = g_0 + g_1 p + \cdots + g_{N-K-1} p^{N-K-1} + p^{N-K}$$

$g(p)$ est le polynôme de degré minimal parmi les 2^K mots de code du code cyclique.

Propriété 3 : Le polynôme $g(p)$ de degré $N - K$ doit être un facteur de $p^N - 1$.

Propriété 4 : L'ensemble des 2^K polynômes associés aux 2^K mots de code d'un code cyclique (N, K) s'obtient par multiplication de $g(p)$ par les 2^K polynômes de degré inférieur ou égal à $K - 1$.

Si on définit le polynôme $u(p)$ associé au mot d'information $\mathbf{u} = [u_0\ u_1\ \ldots\ u_{K-2}\ u_{K-1}]$

$$u(p) = u_0 + u_1 p + \cdots + u_{K-2} p^{K-2} + u_{K-1} p^{K-1}$$

On a la relation suivante entre le polynôme $u(p)$ de degré $K-1$ et le polynôme $c(p)$ de degré $N-1$:

$$c(p) = u(p) g(p) \qquad (9.2)$$

Propriété 5 : Tout polynôme facteur de $p^N - 1$ engendre un code cyclique.

Enumérons la décomposition en produit de polynômes irréductibles des polynômes de la forme $p^{2^j - 1} - 1$ pour $J \leq 5$.

Les polynômes irréductibles de cette table sont les principaux polynômes utilisés pour la construction de codes cycliques. Par exemple $p^7 - 1$ se décompose en un produit de 3 polynômes irréductibles. Il est donc possible de construire les 3 codes cycliques suivants avec $N = 7$:

-un code (7,6) avec $g(p) = 1 + p$
-un code (7,4) avec $g(p) = 1 + p + p^3$
-un code (7,4) avec $g(p) = 1 + p^2 + p^3$

Les codes cycliques (7,4) ainsi obtenus sont des codes de Hamming étudiés précédemment.

$j=2$	$p^3-1=(1+p)(1+p+p^2)$
$j=3$	$p^7-1=(1+p)(1+p+p^3)(1+p^2+p^3)$
$j=4$	$p^{15}-1=(1+p)(1+p+p^2)(1+p^3+p^4)(1+p+p^4)(1+p+p^2+p^3+p^4)$
$j=5$	$p^{31}-1=(1+p)(1+p^3+p^5)(1+p^2+p^5)(1+p^2+p^3+p^4+p^5)$ $(1+p+p^3+p^4+p^5)(1+p+p^2+p^4+p^5)(1+p+p^2+p^3+p^5)$

TAB. 9.1 – Décomposition en produit de polynômes irréductibles des polynômes de la forme $p^{2^j-1}-1$.(table à compléter)

mots d'information				mots de code						
u_1	u_2	u_3	u_4	c_1	c_2	c_3	c_4	c_5	c_6	c_7
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	1	0	1	0	0	0
0	1	0	0	0	1	1	0	1	0	0
1	1	0	0	1	0	1	1	1	0	0
0	0	1	0	0	0	1	1	0	1	0
1	0	1	0	1	1	1	0	0	1	0
0	1	1	0	0	1	0	1	1	1	0
1	1	1	0	1	0	0	0	1	1	0
0	0	0	1	0	0	0	1	1	0	1
1	0	0	1	1	1	0	0	1	0	1
0	1	0	1	0	1	1	1	0	0	1
1	1	0	1	1	0	1	0	0	0	1
0	0	1	1	0	0	1	0	1	1	1
1	0	1	1	1	1	1	1	1	1	1
0	1	1	1	0	1	0	0	0	1	1
1	1	1	1	1	0	0	1	0	1	1

TAB. 9.2 – Liste des mots de code du code de Hamming (7,4) avec $g(p)=1+p+p^3$.

Exemple : Considérons le code de Hamming construit à partir du polynôme générateur $g(p)=1+p+p^3$. Les 16 mots de code s'obtiennent par multiplication des polynômes associés aux mots d'information avec $g(p)$. L'ensemble des 16 mots de code est donné dans la table 9.2.

Soit $\mathbf{u}=[1\ 1\ 1\ 0]$ (en utilisant la convention MSB à droite)
$u(p)=1+p+p^2$
$c(p)=u(p)g(p)=(1+p+p^2)(1+p+p^3)=1+p^4+p^5$
$\mathbf{c}=[1\ 0\ 0\ 0\ 1\ 1\ 0]$

L'obtention de la matrice génératrice à partir du polynôme générateur est immédiate. Nous avons vu précédemment que la matrice génératrice d'un code linéaire en bloc s'obtient à partir de K mots de code indépendants. Dans le cas des codes cycliques, il suffit de choisir les polynômes suivants :

$$g(p) \quad g(p)p \quad g(p)p^2 \quad \dots \quad g(p)p^{K-1}$$

Par exemple, pour le code de Hamming (7,4) avec $g(p)=1+p+p^3$, la matrice génératrice est la suivante :

$$\mathbf{G} = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix} \tag{9.3}$$

La matrice génératrice ainsi obtenue n'est pas sous la forme systématique. En pratique, il est souhaitable que la matrice génératrice soit sous une forme systématique. Soit le mot d'information $\mathbf{u} = [u_0\ u_1\ \ldots\ u_{K-2}\ u_{K-1}]$ et $u(p) = u_0 + u_1 p + \cdots + u_{K-2}p^{K-2} + u_{K-1}p^{K-1}$ son polynôme associé. Multiplions $u(p)$ par p^{N-K} :

$$p^{N-K}u(p) = u_0 p^{N-K} + u_1 p^{N-K+1} + \cdots + u_{K-2}p^{N-2} + u_{K-1}p^{N-1}$$

Le polynôme $c(p)$ associé à un mot de code sous la forme systématique $\mathbf{c} = [c_0\ c_1\ \ldots\ c_{N-1}] = [c_0\ c_1\ \ldots\ c_{N-K-1}\ u_0\ u_1\ \ldots\ u_{K-2}\ u_{K-1}]$ s'écrit :

$$c(p) = r(p) + p^{N-K}u(p) \tag{9.4}$$

$$= q(p)g(p) \qquad \text{d'après la propriété 3 des codes cycliques} \tag{9.5}$$

Divisons $p^{N-K}u(p)$ par $g(p)$. On obtient :

$$p^{N-K}u(p) = q(p)g(p) + r(p)$$

où $q(p)$ est le quotient et $r(p)$ est le reste de la division de degré inférieur à $N-K$.

En résumé :

Pour obtenir le mot de code sous forme systématique $c(p) = q(p)g(p)$ il suffit donc :
-1 de multiplier le polynôme $u(p)$ par p^{N-K}
-2 de diviser $p^{N-K}u(p)$ par $g(p)$ pour obtenir le reste $r(p)$
-3 d'ajouter le reste $r(p)$ à $p^{N-K}u(p)$

$$c(p) = r(p) + p^{N-K}u(p)$$

exemple : Considérons le mot d'information $\mathbf{u} = [1\ 1\ 1\ 0]$ et le code de Hamming (7,4) avec $g(p) = 1 + p + p^3$ Le polynôme associé à $\mathbf{u}$ est $u(p) = 1 + 1p + 1p^2 + 0p^3$. On a :

$$p^3u(p) = p^3 + p^4 + p^5$$

$$\begin{array}{l|l} p^5 + p^4 + p^3 & p^3 + p + 1 \\ \underline{p^5 + p^3 + p^2} & \\ p^4 + p^2 & p^2 + p \\ \underline{p^4 + p^2 + p} & \\ p & \end{array}$$

Le reste de la division de $p^3 + p^4 + p^5$ par $g(p) = 1 + p + p^3$ est égal à p. Ainsi le polynôme $c(p)$ associé au mot de code s'écrit :

$$c(p) = p + p^3 + p^4 + p^5$$

Le mot de code est donc finalement :

$$\begin{array}{ll} \mathbf{c} = [0\ 1\ 0 & 1\ 1\ 1\ 0] \\ c_0c_1c_2 & u_0u_1u_2u_3 \\ c_0c_1c_2 & c_3c_4c_5c_6 \\ \text{contrôle} & \text{information} \end{array}$$

Attention, cette écriture est différente de la forme systématique présentée dans le chapitre sur les codes en blocs linéaires $\mathbf{c} = [c_0c_1c_2c_3c_4c_5c_6] = [u_0u_1u_2u_3c_4c_5c_6]$

9.2 Détection d'erreurs par CRC

Pour détecter la présence d'erreurs dans un mot reçu, la majorité des protocoles utilent une technique baptisée *Cyclic Redundancy Check* (CRC). Cette technique consiste à utiliser un code cyclique.

Il faut souligner que cette technique ne permet pas de corriger les erreurs. On peut cependant demander une réémission du mot de code *Automatic Repeat Request* (ARQ) en anglais.

Soit $g(p)$ le polynôme générateur du CRC. A partir d'un mot d'information $u(p)$, le mot de code $c(p)$ est construit sous forme systématique comme précédemment :

-1 multiplication de $u(p)$ par p^{N-K}

-2 division de $p^{N-K}u(p)$ par $g(p)$ pour obtenir le reste $r(p)$

-3 ajout de $r(p)$ à $p^{N-K}u(p)$

On obtient donc le mot de code systématique $c(p)$ sous la forme suivante :

$$c(p) = r(p) + p^{N-K}u(p)$$

A la réception, on divise le mot reçu par $g(p)$. Si le reste de la division est nul, alors il n'y a pas d'erreur. Sinon, il y a une ou plusieurs erreurs dans le mot reçu.

La table (9.2) donne quelques polynômes utilisés pour la transmission des trames composées d'octets.

CRC16	$g(p) = p^{16} + p^{15} + p^2 + 1$
CRC-CCITT	$g(p) = p^{16} + p^{12} + p^5 + 1$

TAB. 9.3 – polynômes générateurs pour CRC

Le CRC-CCITT est utilisé dans les contrôleurs de disque ainsi que dans les protocoles SDLC,HDLC et X25.

9.3 Codage des codes cycliques

Dans ce paragraphe nous allons étudier l'architecture nécessaire pour le codage des codes cycliques. Nous verrons que ces opérations utilisent de simples additionneurs modulo 2 et des registres à décalage.

9.3.1 Structure matérielle d'un diviseur de polynômes

Soit la division d'un polynôme dividende $a(p)$ par un polynôme diviseur $g(p)$ de degré d dans le corps de Galois GF(2). Le reste $r(p)$ et le quotient $q(p)$ de cette division sont définis par la relation classique :

$$a(p) = g(p)q(p) + r(p)$$

Exemple : Soit la division de $1 + p^2 + p^6$ par $1 + p + p^2$.

En analysant les étapes du calcul de cette division, on peut faire les remarques suivantes :

-1 Au cours de la division, le degré du dividende décroît : les modifications du dividende sont réalisées de la gauche vers la droite (des poids forts vers les poids faibles)

-2 Pour un diviseur de degré d, à chaque itération, les modifications du dividende ne portent que sur les $(d+1)$ termes les plus à gauche. Les autres termes du dividende peuvent être momentanément ignorés.

-3 Lorsque la modification du dividende a lieu, elle consiste à ajouter terme à terme les $d+1$ coefficients du diviseur.

$$
\begin{array}{l|l}
1p^6+0p^5+0p^4+0p^3+1p^2+0p+1 & 1p^2+1p+1 \\
\underline{1p^6+1p^5+1p^4} & \\
0p^6+1p^5+1p^4+0p^3 & 1p^4+1p^3+0p^2+1p+0 \\
\quad \underline{1p^5+1p^4+1p^3} & \\
\quad 0p^5+0p^4+1p^3+1p^2 & \\
\quad\quad \underline{0p^4+0p^3+0p^2} & \\
\quad\quad 0p^4+1p^3+1p^2+0p & \\
\quad\quad\quad \underline{1p^3+1p^2+1p} & \\
\quad\quad\quad 0p^3+0p^2+1p+1 &
\end{array}
$$

Fort de ces remarques, il est possible de déduire une structure matérielle pour réaliser cette division. Cette structure comporte autant de registres à décalage que le degré du diviseur. Elle est présentée sur la figure 9.1 pour l'exemple traité ci-dessus.

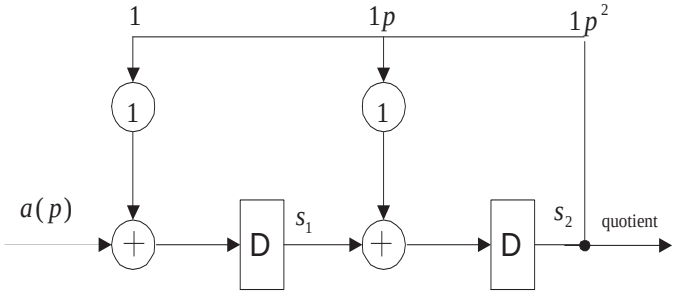


FIG. 9.1 – structure matérielle d'un diviseur par $1 + p + p^2$

Il est important de préciser que les bits entrent dans le diviseur poids fort (MSB) en tête.

A chaque coup d'horloge, un nouveau coefficient du polynôme $a(p)$ entre dans le circuit. Après d coups d'horloge, le premier coefficient non nul du quotient sort du dernier registre à décalage. Ce coefficient est multiplié par $g(p)$ puis soustrait comme dans une division classique.

Le séquencement correspondant à la division calculée ci-dessus est donné sur la figure 9.2 et dans la table 9.5. On peut vérifier qu'à chaque instant les registres internes contiennent la partie du dividende en cours de calcul. Le quotient est obtenu à partir du second coup d'horloge en sortie du registre s_2 $q(p) = 1p^4 + 1p^3 + 0p^2 + 1p + 0$. Au dernier coup d'horloge, la sortie des registres à décalage correspond bien au reste de la division soit $r(p) = 1p + 1$.

9.3.2 Structures d'un codeur cyclique

Dans le cas des codes cycliques (N, K), nous avons vu qu'avant de calculer le reste $r(p)$ de la division il est nécessaire de prémultiplier le polynôme associé au mot d'information $u(p)$ par p^{N-K}. La multiplication par p^{N-K} est équivalente à ajouter p^{N-K} zéros à la suite de $u(p)$.

exemple : Considérons à nouveau l'exemple du code cyclique de Hamming (7,4) avec $g(p) = 1 + p + p^3$

La structure du codeur est donnée figure 9.3. Comme précédemment les bits sont introduits dans le codeur MSB en tête.

Reprenons le mot d'information $\mathbf{u} = [1\ 1\ 1\ 0]$ (en utilisant toujours la convention MSB à droite) associé au polynôme $u(p) = 1 + p + p^2$

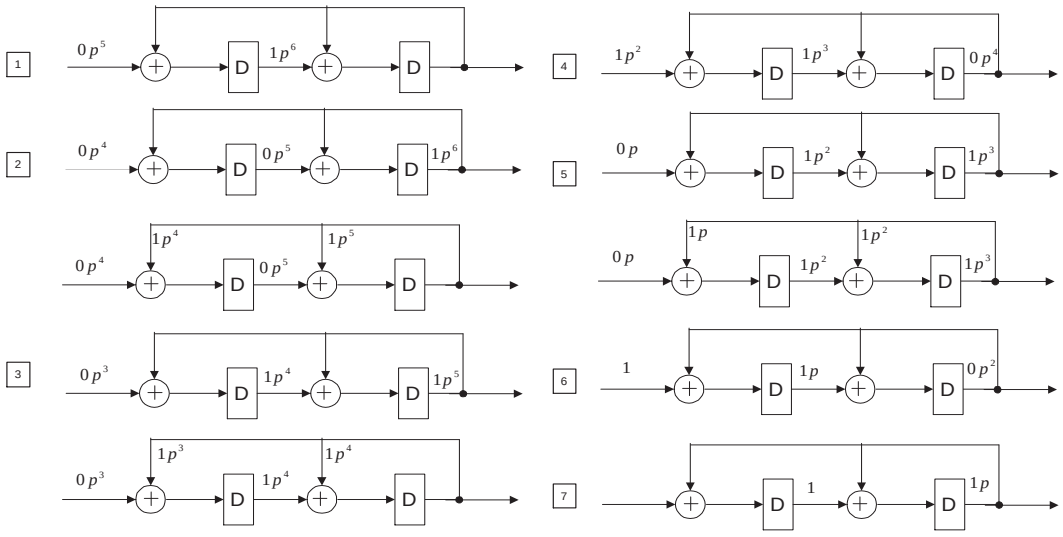


FIG. 9.2 – séquencement du diviseur par $1 + p + p^2$

	entrée	coup d'horloge	s_1	s_2
		0	0	0
MSB	1	1	1	0
	0	2	0	1
	0	3	1	1
	0	4	1	0
	1	5	1	1
	0	6	1	0
LSB	1	7	1	1

TAB. 9.4 – séquencement du diviseur

Après prémultiplication, le mot transmis au diviseur est [0 0 0 1 1 1 0]

On retrouve le reste calculé précédemment soit $r(p) = 0 + 1p + 0p^2$. Avec cette structure, il faut 7 coups d'horloge pour obtenir le reste de la division sur 3 bits.

Il est possible de réduire le nombre de coups d'horloge nécessaire pour calculer le reste en exploitant le fait que les $N - K$ derniers bits du polynôme $p^{N-K}u(p)$ sont nuls. Au lieu de modifier les $N - K + 1$ coefficients les plus à gauche du dividende à chaque itération, on peut décomposer la division en K divisions successives.

Reprenons l'exemple précédent pour illustrer ce principe : $u(p) = 1+p+p^2$ et $g(p) = 1+p+p^3$.

La structure du codeur correspondant est donnée sur la figure 9.4.

En résumé, cette structure permet de gagner $N-K$ coups d'horloge correspondant aux $N-K$ derniers bits nuls du polynôme $p^{N-K}u(p)$.

Nous pouvons maintenant présenter la structure matérielle complète d'un codeur pour le code de Hamming (7,4) défini par le polynôme $g(p) = 1 + p + p^3$

Le codage s'effectue en deux étapes :

1/ L'interrupteur P_0 est fermé et P_1 est en position A. On applique $K = 4$ coups d'horloge pour envoyer les K premiers bits du mot de code et calculer le reste $r(p)$.

2/ L'interrupteur P_0 est ouvert et P_1 est en position B. On applique $N-K = 3$ coups d'horloge pour envoyer les $N - K$ derniers bits du mot de code (correspondant au reste).

Ces structures matérielles de codeurs sont à comparer à celles utilisant des additionneurs

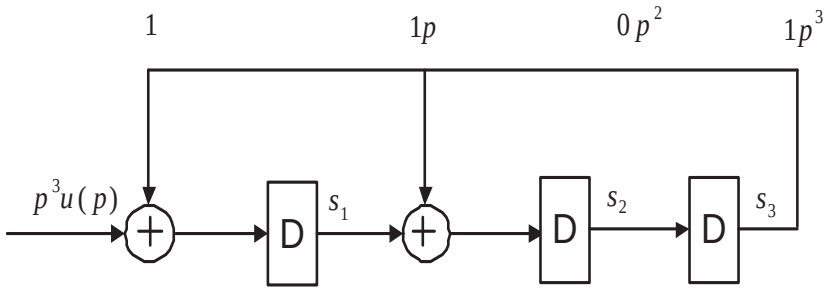


FIG. 9.3 – structure matérielle d'un codeur de Hamming $g(p) = 1 + p + p^3$

	entrée	coup d'horloge	s_1	s_2	s_3
		0	0	0	0
MSB	0	1	0	0	0
	1	2	1	0	0
	1	3	1	1	0
	1	4	1	1	1
	0	5	1	0	1
	0	6	1	0	0
LSB	0	7	0	1	0

TAB. 9.5 – séquencement du codeur de Hamming

modulo 2 définies à partir du graphe de Tanner.

9.4 Décodage des codes cycliques

Le décodage d'un code cyclique en bloc comprend deux phases distinctes :
-1 calcul de syndrome
-2 localisation des erreurs

Le calcul du syndrome s'obtient en réalisant la division du mot reçu $y(p)$ par $g(p)$.

$$y(p) = q(p)g(p) + s(p) \tag{9.6}$$

$$\begin{array}{l|l}
1p^5 & 1p^3 + 0p^2 + 1p + 1 \\
+\ 1p^5 + 0p^4 + 1p^3 + 1p^2 & \overline{} \\
\overline{0p^5 + 0p^4 + 1p^3 + 1p^2} & 1p^2 + 1p \\
+\ \ 1p^4 & \\
\overline{1p^4 + 1p^3 + 1p^2} & \\
+\ \ 1p^4 + 0p^3 + 1p^2 + 1p & \\
\overline{1p^3 + 0p^2 + 1p} & \\
+\ \ 1p^3 & \\
\overline{1p} &
\end{array}$$

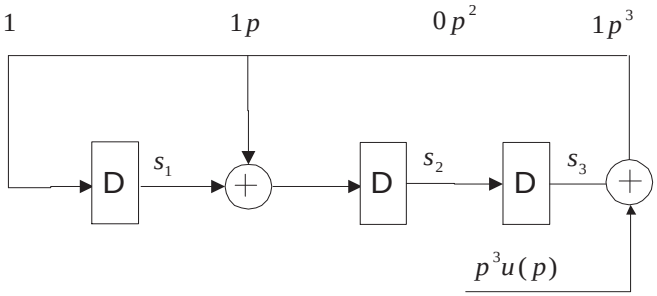


FIG. 9.4 – structure matérielle d'un codeur de Hamming $g(p) = 1 + p + p^3$

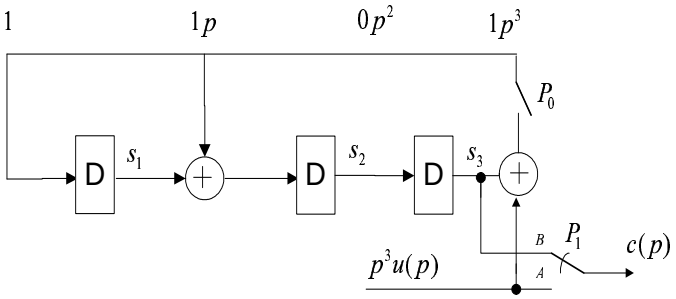


FIG. 9.5 – structure matérielle complète d'un codeur de Hamming $g(p) = 1 + p + p^3$

Le reste de cette division possède toutes les propriétés d'un syndrome :
Si $s(p) = 0$, alors le mot reçu $y(p)$ est un mot de code
$s(p)$ est un polynôme de degré inférieur ou égal à $N - K - 1$. A partir de la valeur de $s(p)$ il est possible d'estimer un vecteur d'erreur $\hat{e}(p)$.

Sur la figure 9.6 nous présentons la structure d'un décodeur relatif au code de Hamming (7,4) défini par le polynôme $g(p) = 1 + p + p^3$

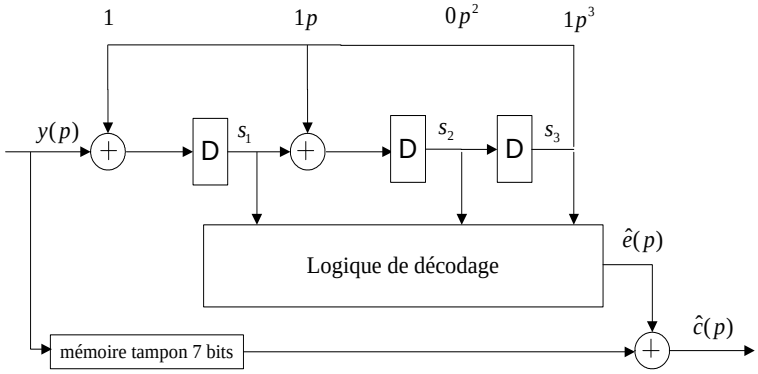


FIG. 9.6 – structure du décodeur pour le code de Hamming $g(p) = 1 + p + p^3$

La logique de décodage associe à chaque valeur du syndrome $s(p)$ le vecteur d'erreur estimé $\hat{e}(p)$. Il suffit ensuite d'ajouter ce vecteur d'erreur au mot reçu $y(p)$:

$$\hat{c}(p) = y(p) + \hat{e}(p) \tag{9.7}$$

La complexité de ce décodeur réside dans la logique de décodage. Cette solution est cependant intéressante lorsque le nombre d'erreurs à corriger est faible.

9.4.1 Correction d'une erreur

Lorsque le code permet de corriger une seule erreur, la logique de décodage est une simple porte à (N-K) entrées.

Soit une erreur de la forme $e(p) = p^j$. Le syndrome associé à cette erreur a la forme particulière suivante :

$$s(p) = p^j \mod g(c) \tag{9.8}$$

Sur la figure 9.6 nous présentons cette structure de décodage pour le code de Hamming (7,4) défini par le polynôme $g(p) = 1 + p + p^3$

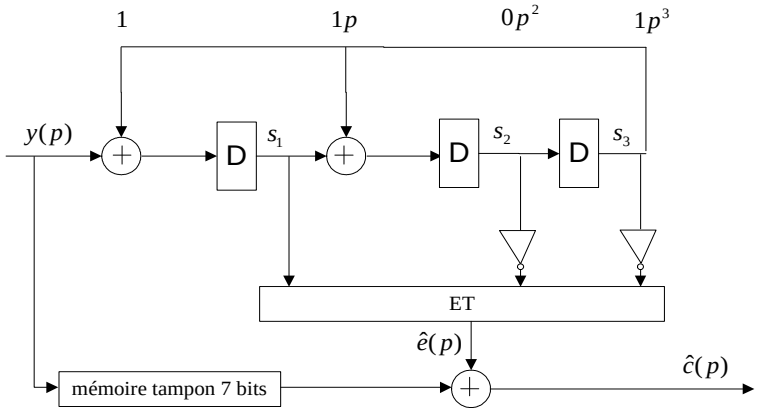


FIG. 9.7 – structure de décodage à complexité réduite pour le code de Hamming défini par $g(p) = 1 + p + p^3$

Pour cet exemple, nous donnons table 9.6, la table de correspondance entre les vecteurs d'erreurs et les syndromes $s(p)$.

vecteur d'erreur	$e(p)$	$s(p)$
1000000	1	1
0100000	p	p
0010000	p^2	p^2
0001000	p^3	$1 + p$
0000100	p^4	$p + p^2$
0000010	p^5	$1 + p + p^2$
0000001	p^6	$1 + p^2$
LSB MSB		

TAB. 9.6 – table des syndromes

Le résultat de la multiplication du syndrome $s(p)$ par p^{N-j} est égal à 1 :

$$s(p)p^{N-j} = p^j p^{N-j} = p^N = 1 \tag{9.9}$$

Ainsi pour déterminer la position de l'erreur, il suffit de multiplier successivement $s(p)$ par p jusqu'à obtenir 1.

Exemple : Soit le mot de code émis $\mathbf{c} = [1111111]$ et le vecteur d'erreur $\mathbf{e} = [0000100]$ ($e(p) = p^4$ soit une erreur au cinquième bit à partir du LSB).

$$y(p) = c(p) + e(p) = 1p^6 + 1p^5 + \mathbf{0}p^4 + 1p^3 + 1p^2 + 1p + 1 \tag{9.10}$$

	entrée	coup d'horloge	s_1	s_2	s_3	
		0	0	0	0	
MSB	1	1	1	0	0	
	1	2	1	1	0	
	0	3	0	1	1	
	1	4	0	1	1	
	1	5	0	1	1	
	1	6	0	1	1	
LSB	1	7	0	1	1	$\rightarrow s(p) = p + p^2$
		8	1	1	1	$\rightarrow s(p)p$
		9	1	0	1	$\rightarrow s(p)p^2$
		10	1	0	0	$\rightarrow s(p)p^3 = 1$
						on a donc $j = 7 - 3 = 4$

TAB. 9.7 – Evolution du contenu des registres

élément	polynôme	représentation binaire
0	0	0000
1	1	0001
α	p	0010
α^2	p^2	0100
α^3	p^3	1000
α^4	$1 + p$	0011
α^5	$p + p^2$	0110
α^6	$p^2 + p^3$	1100
α^7	$1 + p + p^3$	1011
α^8	$1 + p^2$	0101
α^9	$p + p^3$	1010
α^{10}	$1 + p + p^2$	0111
α^{11}	$p + p^2 + p^3$	1110
α^{12}	$1 + p + p^2 + p^3$	1111
α^{13}	$1 + p^2 + p^3$	1101
α^{14}	$1 + p^3$	1001

TAB. 9.8 – Liste des éléments du corps GF(2^4).

Suivons l'évolution du contenu des registres du décodeur sur la table 9.7 :
Comme attendu, nous avons pu corriger l'erreur 3 coups d'horloge après avoir calculé le reste.

9.5 Corps de Galois à 2^m éléments

Le corps de Galois fini GF(2^m) est isomorphe au corps des polynômes à coefficients dans GF(2) modulo un polynôme irréductible dans GF(2) et de degré m.

Prenons un exemple pour illustrer ces propos. Soit $m = 4$ et le polynôme irréductible $1+p+p^4$. Nous avons vu dans la table 9.1 que ce polynôme est facteur de $p^{15} - 1$.

Soit α une racine de ce polynôme : $1 + \alpha + \alpha^4 = 0$. On peut montrer que les puissances successives de α engendrent les $2^m - 1$ éléments non nuls du corps GF(2^m). La table 9.8 liste les éléments du corps GF(2^4) ainsi construit.

A chaque élément non nul du corps GF(2^m) on associe un polynôme minimal.

Définition : le polynôme minimal $m_i(p)$ est le polynôme irréductible de plus faible degré dont α^i est racine.

Pour l'exemple précédent les 15 polynômes minimaux sont les suivants :

$$\begin{aligned}
&m_1(p) = m_2(p) = m_4(p) = m_8(p) = 1 + p + p^4 \\
&m_3(p) = m_6(p) = m_9(p) = m_{12}(p) = 1 + p + p^2 + p^3 + p^4 \\
&m_5(p) = m_{10}(p) = 1 + p + p^2 \\
&m_7(p) = m_{11}(p) = m_{13}(p) = m_{14}(p) = 1 + p^3 + p^4 \\
&m_0(p) = 1 + p
\end{aligned} \tag{9.11}$$

On retrouve les 5 polynômes irréductibles facteur de $p^{15} - 1$.

9.6 Les codes BCH

Les codes BCH sont des codes cycliques binaires. Ils ont été découverts par Hocquenghem [1] [15] puis par Bose et Ray-Chaudhuri [4]. Ils permettent de construire des codes avec les paramètres suivants :

$$\begin{aligned}
N &= 2^m - 1 \\
N - K &\leq me \\
d_{min} &\geq 2e + 1
\end{aligned} \tag{9.12}$$

Comme tous les codes cycliques, un code BCH est défini par son polynôme générateur $g(p)$. Le polynôme générateur $g(p)$ possède parmi ses racines les $2e$ puissances consécutives de α. Ainsi, le polynôme générateur $g(p)$ est le produit des polynômes minimaux associés à $\alpha, \alpha^2, \alpha^3, \ldots, \alpha^{2e}$ sans qu'aucun des polynômes ne soit répété :

$$g(p) = \text{PPCM}\Big[m_1(p), m_2(p), m_3(p), \ldots, m_{2e}(p)\Big] \tag{9.13}$$

PPCM : plus petit commun multiple.

Comme dans GF(2^m), α^i et α^{2i} sont racines du même polynôme minimal, pour la détermination de $g(p)$, il suffit de ne considérer que les puissances impaires de α :

$$g(p) = \text{PPCM}\Big[m_1(p), m_3(p), \ldots, m_{2e-1}(p)\Big] \tag{9.14}$$

Exemple : Construisons un code BCH avec $N = 15$ et sachant corriger deux erreurs soit $e = 2$. Puisque $N = 2^m - 1$, on obtient $m = 4$ et la distance minimale est supérieure ou égale à 5. Son polynôme générateur $g(p)$ est le suivant :

$$g(p) = m_1(p)m_3(p)$$

A partir de la liste des polynômes minimaux pour GF(2^4), on obtient :

$$g(p) = (1 + p + p^4)(1 + p + p^2 + p^3 + p^4) = 1 + p^4 + p^6 + p^7 + p^8$$

Comme le degré de $g(p)$ est $N - K = 8$, le code BCH ainsi généré sera un code (15,7). Sa distance minimale est exactement égale à 5.

La table 9.9 présente les polynômes générateurs des codes BCH corrigeant jusqu'à 3 erreurs avec $N \leq 63$. Une table plus exhaustive est donnée dans [23].

[1] A Hocquenghem était professeur de mathématiques au CNAM

	N	K	e	$g(p)$
GF(2^3)	7	4	1	p^3+p+1
GF(2^4)	15	11	1	p^4+p+1
		7	2	$p^8+p^7+p^6+p^4+1$
		5	3	$p^{10}+p^8+p^5+p^4+p^2+p+1$
GF(2^5)	31	26	1	p^5+p^2+1
		21	2	$p^{10}+p^9+p^8+p^6+p^5+p^3+1$
		16	3	$p^{15}+p^{11}+p^{10}+p^9+p^8+p^7+p^5+p^3+p^2+p+1$
GF(2^6)	63	57	1	p^6+p+1
		51	2	$p^{12}+p^{10}+p^8+p^5+p^4+p^3+1$
		45	3	$p^{18}+p^{17}+p^{16}+p^{15}+p^9+p^7+p^6+p^3+p^2+p+1$

TAB. 9.9 – Liste des polynômes générateurs des codes BCH avec $N \leq 63$.

Nous savons que le polynôme générateur $g(p)$ d'un code BCH possède $\alpha, \alpha^2, \ldots, \alpha^{2e}$ comme racines. Ainsi tout mot de code aura aussi $\alpha, \alpha^2, \alpha^3, \ldots, \alpha^{2e}$ comme racines.

Soit un mot de code $c(p) = c_0 + c_1 p + \cdots + c_{N-1}p^{N-1}$

Cette propriété peut s'écrire comme suit :

$$c(\alpha^i) = c_0 + c_1\alpha^i + \cdots + c_{N-1}\alpha^{(N-1)i} = 0 \quad \forall i \quad 1 \leq i \leq 2e \tag{9.15}$$

Cette relation peut s'écrire aussi sous la forme matricielle suivante :

$$(c_0 \quad c_1 \ldots c_{N-1}) \begin{pmatrix} 1 & 1 & \ldots & 1 \\ \alpha & \alpha^2 & \ldots & \alpha^{2e} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha^{N-1} & \alpha^{2(N-1)} & \ldots & \alpha^{2e(N-1)} \end{pmatrix} = 0 \tag{9.16}$$

La matrice de parité s'écrit donc :

$$\mathbf{H} = \begin{pmatrix} 1 & \alpha & \ldots & \alpha^{N-1} \\ 1 & \alpha^2 & \ldots & \alpha^{2(N-1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^{2e} & \ldots & \alpha^{2e(N-1)} \end{pmatrix} \tag{9.17}$$

Cette matrice vérifie la relation $\mathbf{cH}^T = 0$. Il est alors possible de démontrer que le code corrige bien e erreurs.

Exemple (suite) :

Les lignes de la matrice de parité (9.6) qui correspondent à un même polynôme minimal n'ont pas besoin d'être répétées et on obtient donc pour le code BCH défini par $g(p) = 1+p^4+p^6+p^7+p^8$ la matrice de parité suivante :

$$\mathbf{H} = \begin{pmatrix} 1 & \alpha & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 & \alpha^7 & \alpha^8 & \alpha^9 & \alpha^{10} & \alpha^{11} & \alpha^{12} & \alpha^{13} & \alpha^{14} \\ 1 & \alpha^3 & \alpha^6 & \alpha^9 & \alpha^{12} & 1 & \alpha^3 & \alpha^6 & \alpha^9 & \alpha^{12} & 1 & \alpha^3 & \alpha^6 & \alpha^9 & \alpha^{12} \end{pmatrix} \tag{9.18}$$

Finalement, en remplaçant chaque puissance de α par sa représentation binaire, on obtient :

$$\mathbf{H} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \tag{9.19}$$

9.6.1 Décodage des codes BCH

9.6.2 Introduction

Le décodage classique des codes BCH peut donc être séparé en 3 étapes distinctes :

1. Calcul des $2e$ composantes du syndrome $\mathbf{s} = (s_1 s_2 \dots s_{2e})$
2. Recherche du polynôme localisateur d'erreurs $\sigma(p)$
3. Détermination des racines du polynôme $\sigma(p)$

9.6.3 calcul du syndrome

Soit $c(p) = c_0 + c_1 p + \cdots + c_{N-2}p^{N-2} + c_{N-1}p^{N-1}$ le mot de code, $y(p) = y_0 + y_1 p + \cdots + y_{N-2}p^{N-2} + y_{N-1}p^{N-1}$ le mot reçu et $e(p) = e_0 + e_1 p + \cdots + e_{N-2}p^{N-2} + e_{N-1}p^{N-1}$ le mot d'erreur. On a alors :

$$y(p) = c(p) + e(p) \tag{9.20}$$

La première étape consiste à calculer le syndrome $\mathbf{s}$ défini par un vecteur à $2e$ éléments. Par définition le syndrome est égal au produit du vecteur reçu par la matrice de controle transposé :

$$\mathbf{s} = (s_1, s_2, \dots, s_{2e}) = \mathbf{y}\mathbf{H}^T = (y_0, y_1, \dots, y_{N-1})\,\mathbf{H}^T \tag{9.21}$$

Les $2e$ composantes du syndrome sont calculées comme suit :

$$s_i = y(\alpha^i) = c(\alpha^i) + e(\alpha^i) = e(\alpha^i) \quad \forall i \quad 1 \le i \le 2e \tag{9.22}$$

Il faut noter que si il n'y a pas d'erreur ($e(p) = 0$), les $2e$ composantes du syndrome sont nulles.
Supposons que le mot d'erreur soit composé de v erreurs (avec $v \le e$) :

$$e(p) = p^{j_1} + p^{j_2} + \cdots + p^{j_v} \tag{9.23}$$

avec $0 \le j_1 \le j_2 \cdots \le j_v \le N-1$.
A partir de (9.6.3), on obtient le système d'équation suivant :

$$\begin{aligned} s_1 &= e(\alpha) = \alpha^{j_1} + \alpha^{j_2} + \cdots + \alpha^{j_v} \\ s_2 &= e(\alpha^2) = (\alpha^{j_1})^2 + (\alpha^{j_2})^2 + \cdots + (\alpha^{j_v})^2 \\ s_3 &= e(\alpha^3) = (\alpha^{j_1})^3 + (\alpha^{j_2})^3 + \cdots + (\alpha^{j_v})^3 \\ &\vdots \\ s_{2e} &= e(\alpha^{2e}) = (\alpha^{j_1})^{2e} + (\alpha^{j_2})^{2e} + \cdots + (\alpha^{j_v})^{2e} \end{aligned} \tag{9.24}$$

Posons $\beta_k = \alpha^{j_k} \quad 1 \le k \le v$ le système d'équation devient :

$$\begin{aligned} s_1 &= \beta_1 + \beta_2 + \cdots + \beta_v \\ s_2 &= (\beta_1)^2 + (\beta_2)^2 + \cdots + (\beta_v)^2 \\ s_3 &= (\beta_1)^3 + (\beta_2)^3 + \cdots + (\beta_v)^3 \\ &\vdots \\ s_{2e} &= (\beta_1)^{2e} + (\beta_2)^{2e} + \cdots + (\beta_v)^{2e} \end{aligned} \tag{9.25}$$

Définissons le polynôme suivant :

$$\begin{aligned} \sigma(p) &= \prod_{k=1}^{v} (1 - \beta_k p) \\ &= \sigma_0 + \sigma_1 p + \sigma_2 p^2 + \ldots \sigma_v p^v \end{aligned} \tag{9.26}$$

Ce polynôme est appelé polynôme localisateur d'erreur car les inverses des racines de $\sigma(p)$ sont égales à $\beta_k = \alpha^{j_k}$ et permettent de localiser la position des erreurs j_k.

Les coefficients σ_k vérifient les équations suivantes :

$$\begin{aligned} \sigma_0 &= 1 \\ \sigma_1 &= \beta_1 + \beta_2 + \cdots + \beta_v \\ \sigma_2 &= \beta_1\beta_2 + \beta_1\beta_3 + \cdots + \beta_{v-1}\beta_v \\ &\vdots \\ \sigma_v &= \beta_1\beta_2 \ldots \beta_v \end{aligned} \tag{9.27}$$

Pour déterminer les coefficients σ_k, on utilisera les relations dites de Newton suivantes :

$$\begin{aligned} &s_1 + \sigma_1 = 0 \\ &s_2 + \sigma_1 s_1 + 2\sigma_2 = 0 \\ &s_3 + \sigma_1 s_2 + \sigma_2 s_1 + 3\sigma_3 = 0 \\ &\vdots \\ &s_v + \sigma_1 s_{v-1} + \cdots + \sigma_{v-1} s_1 + v\sigma_v = 0 \end{aligned} \tag{9.28}$$

et pour $i > v$

$$s_i + \sigma_1 s_{i-1} + \cdots + \sigma_v s_{i-v} = 0$$

Il faut noter que dans le cas binaire $i\sigma_i = 0$ si i est pair.

9.6.4 Détermination du polynôme localisateur d'erreur par l'approche matricielle

L' approche matricielle consiste à résoudre directement le système de Newton (9.6.3) pour déterminer le polynôme localisateur d'erreur $\sigma(p)$.

Dans le cas binaire, le système de Newton (9.6.3) peut s'écrire sous la forme matricielle suivante :

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & \dots & 0 \\ s_2 & s_1 & 1 & 0 & 0 & \dots & 0 \\ s_4 & s_3 & s_2 & s_1 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ s_{2v-4} & s_{2v-5} & \dots & \dots & \dots & \dots & s_{v-3} \\ s_{2v-2} & s_{2v-3} & \dots & \dots & \dots & \dots & s_{v-1} \end{pmatrix} \begin{pmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \\ \vdots \\ \sigma_{v-1} \\ \sigma_v \end{pmatrix} = \begin{pmatrix} s_1 \\ s_3 \\ s_5 \\ \vdots \\ s_{2v-3} \\ s_{2v-1} \end{pmatrix} \tag{9.29}$$

Par exemple si $v = 2$ le système devient

$$\begin{pmatrix} 1 & 0 \\ s_2 & s_1 \end{pmatrix} \begin{pmatrix} \sigma_1 \\ \sigma_2 \end{pmatrix} = \begin{pmatrix} s_1 \\ s_3 \end{pmatrix} \tag{9.30}$$

Et les solutions sont :
$\sigma_1 = s_1 \qquad \sigma_2 = \frac{s_3}{s_1} + s_2.$

La procédure par approche matricielle peut être résumée ainsi :

Tout d'abord on fait l'hypothèse que $v = e$ et on essaye de résoudre le système de Newton (9.6.4) en remplaçant v par e. Si v est égal à e ou $e-1$, la solution unique sera trouvée. Si $v < e-1$, alors le système n'aura pas de solution. Dans ce cas, on doit faire l'hypothèse que $v = e - 2$ et essayer de résoudre le système de Newton (9.6.4) en remplaçant cette fois ci v par $e-2$. On répète cette technique jusqu'à ce que l'on trouve une solution.

On peut se rendre compte que l'approche matricielle ne présente un interet que lorsque la capacité de correction du code BCH est limitée (par exemple $e \leq 3$).

Nous allons maintenant voir un algorithme dont la portée est beaucoup plus générale : l'algorithme de Berlekamp-Massey.

9.6.5 Détermination du polynôme localisateur d'erreur par l'algorithme de Berlekamp-Massey

Cet algorithme a été proposé par Berlekamp [2] et son fonctionnement a été éclairci par Massey [22]. Nous nous contenterons ici de présenter l'algorithme sans en faire la démonstration. Pour une description détaillée, nous renvoyons le lecteur à l'ouvrage de Berlekamp.

La première étape consiste à déterminer le polynôme de degré minimal $\sigma^{(1)}(p)$ dont les coefficients vérifient la première égalité de Newton (9.6.3). Si les coefficients vérifient aussi la seconde égalité de Newton, on pose alors $\sigma^{(2)}(p) = \sigma^{(1)}(p)$. Sinon, on ajoute un terme correctif à $\sigma^{(1)}(p)$ pour obtenir $\sigma^{(2)}(p)$. Le polynôme $\sigma^{(2)}(p)$ est alors le polynôme à degré minimum dont les coefficients vérifient les deux premières égalités de Newton. On continue cette procédure pour former $\sigma^{(3)}(p)$ puis $\sigma^{(4)}(p)$ jusqu'à obtenir $\sigma^{(2e)}(p)$. Ce dernier polynôme est le polynôme localisateur $\sigma(p)$.

Soit $\sigma^{(\mu)}(p) = 1 + \sigma_1^{(\mu)} p + \sigma_2^{(\mu)} p^2 + \cdots + \sigma_{l_\mu}^{(\mu)} p^{l_\mu}$ le polynôme de degré l_μ déterminé à la μ^{eme} étape et vérifiant donc les μ premières équations. Pour déterminer $\sigma^{(\mu+1)}(p)$ on applique la procédure suivante :

on commence par calculer d_μ comme suit :

$$d_\mu = s_{\mu+1} + \sigma_1^{(\mu)} s_\mu + \sigma_2^{(\mu)} s_{\mu-1} + \cdots + \sigma_{l_\mu}^{(\mu)} s_{\mu+1-l_\mu}$$

Si $d_\mu = 0$, alors $\sigma^{(\mu+1)}(p) = \sigma^{(\mu)}(p)$. Sinon, on remonte à l'itération ρ telle que $d_\rho \neq 0$ et $\rho - l_\rho$ soit maximal (avec l_ρ degré du polynôme $\sigma^{(\rho)}(p)$). On obtient alors :

$$\sigma^{(\mu+1)}(p) = \sigma^{(\mu)}(p) + d_\mu d_\rho^{-1} p^{\mu-\rho} \sigma^{(\rho)}(p)$$

Exemple (suite) :

Considérons à nouveau le code BCH (15,7) défini par le polynôme générateur $g(p) = 1 + p^4 + p^6 + p^7 + p^8$ qui peut corriger jusqu'à 2 erreurs.

Supposons que le mot de code tout à zéro $x(p) = 0$ a été transmis et que le mot reçu est égal à $r(p) = p^4 + p^9$

On commence par calculer les composantes du syndrome

$$\begin{aligned} s_1 &= r(\alpha) = \alpha^4 + \alpha^9 = 1 + \alpha^3 = \alpha^{14} \\ s_2 &= r(\alpha^2) = \alpha^8 + \alpha^{18} = 1 + \alpha^2 + \alpha^3 = \alpha^{13} \\ s_3 &= r(\alpha^3) = \alpha^{12} + \alpha^{27} = 0 \\ s_4 &= r(\alpha^4) = \alpha^{16} + \alpha^{36} = \alpha + \alpha^2 + \alpha^3 = \alpha^{11} \end{aligned} \tag{9.31}$$

Comme l'exemple est relativement simple, il est possible de déterminer le polynôme localisateur d'erreur $\sigma(p)$ par l'approche matricielle :

On obtient :

$$\begin{aligned} \sigma_1 &= s_1 = \alpha^{14} \\ \sigma_2 &= \frac{s_3}{s_1} + s_2 = s_2 = \alpha^{13} \end{aligned} \tag{9.32}$$

Utilisons maintenant l'algorithme de Berlekamp-Massey :

- $\mu = 0$

On initialise $\sigma^{(0)}(p) = 1$ et on calcule d_0 :

$$d_0 = s_1 = \alpha^{14}$$

Nous obtenons le tableau suivant :

TAB. 9.10 – tableau partiel pour $\mu = 0$

μ	$\sigma^{(\mu)}(p)$	d_μ	l_μ	$\mu - l_\mu$
-1	1	1	0	-1
0	1	α^{14}	0	0

Comme d_0 est non nul, nous devons ajouter un terme correctif à $\sigma^{(0)}(p)$ pour obtenir $\sigma^{(1)}(p)$:

$$\begin{aligned} \sigma^{(1)}(p) &= \sigma^{(0)}(p) + d_\mu d_\rho^{-1} p^{\mu-\rho} \sigma^{(\rho)}(p) \quad \text{avec} \quad \rho = -1 \\ &= 1 + \alpha^{14} p \end{aligned} \tag{9.33}$$

- $\mu = 1$

On calcule d_1 :

$$\begin{aligned} d_1 &= s_2 + \sigma_1^{(\mu)} s_1 \\ &= \alpha^{13} + \alpha^{14}\alpha^{14} \\ &= \alpha^{13} + \alpha^{13} = 0 \end{aligned} \tag{9.34}$$

TAB. 9.11 – tableau partiel pour $\mu = 1$

μ	$\sigma^{(\mu)}(p)$	d_μ	l_μ	$\mu - l_\mu$
-1	1	1	0	-1
0	1	α^{14}	0	0
1	$1 + \alpha^{14}p$	0	1	0

Nous obtenons le tableau suivant :

Comme d_1 est nul, on a :

$$\sigma^{(2)}(p) = \sigma^{(1)}(p) = 1 + \alpha^{14}p$$

- $\mu = 2$

On calcule d_2 :

$$\begin{aligned} d_2 &= s_3 + \sigma_1^{(\mu)} s_2 + \sigma_2^{(\mu)} s_1 \\ &= 0 + \alpha^{13}\alpha^{14} + 0 \\ &= \alpha^{12} \end{aligned} \tag{9.35}$$

Nous obtenons le tableau suivant :

TAB. 9.12 – tableau partiel pour $\mu = 2$

μ	$\sigma^{(\mu)}(p)$	d_μ	l_μ	$\mu - l_\mu$
-1	1	1	0	-1
0	1	α^{14}	0	0
1	$1 + \alpha^{14}p$	0	1	0
2	$1 + \alpha^{14}p$	α^{12}	1	1

Comme d_2 est non nul, nous devons ajouter un terme correctif à $\sigma^{(2)}(p)$ pour obtenir $\sigma^{(3)}(p)$:

$$\begin{aligned} \sigma^{(3)}(p) &= \sigma^{(2)}(p) + d_\mu d_\rho^{-1} p^{\mu-\rho}\sigma^{(\rho)}(p) \quad \text{avec} \quad \rho = 0 \\ &= 1 + \alpha^{14}p + \alpha^{12}\alpha^{-14}p^2 \\ &= 1 + \alpha^{14}p + \alpha^{13}p^2 \end{aligned} \tag{9.36}$$

- $\mu = 3$

On calcule d_3 :

$$\begin{aligned} d_3 &= s_4 + \sigma_1^{(\mu)} s_3 + \sigma_2^{(\mu)} s_2 \\ &= \alpha^{11} + 0 + \alpha^{13}\alpha^{13} \\ &= 0 \end{aligned} \tag{9.37}$$

Nous obtenons le tableau suivant :

TAB. 9.13 – tableau final

μ	$\sigma^{(\mu)}(p)$	d_μ	l_μ	$\mu - l_\mu$
-1	1	1	0	-1
0	1	α^{14}	0	0
1	$1+\alpha^{14}p$	0	1	0
2	$1+\alpha^{14}p$	α^{12}	1	1
3	$1+\alpha^{14}p+\alpha^{13}p^2$	0	2	1

Comme d_3 est nul, on a $\sigma^{(4)}(p) = \sigma^{(3)}(p)$ et on obtient finalement le polynôme localisateur d'erreur $\sigma(p)$ suivant :

$$\sigma(p) = \sigma^{(2e)}(p) = \sigma^{(4)}(p) = 1 + \alpha^{14}p + \alpha^{13}p^2$$

Il reste maintenant à déterminer les positions des erreurs.

9.6.6 Recherche des racines du polynôme localisateur d'erreur

Nous avons vu précédemment que les inverses des racines du polynôme localisateur d'erreur sont égales à β_k et permettent de localiser la position des erreurs j_k.

Si le degré de $\sigma(p)$ est égal à 1 ou 2, les racines peuvent être déterminées directement. Lorsque le degré de $\sigma(p)$ est supérieur à 2, on peut faire le test de chaque puissance de α pour voir si c'est une racine de $\sigma(p)$. Une autre solution plus simple consiste à appliquer la méthode de Chien :

- **initialisation**

$$Q_k = \sigma_k \quad \text{avec} \quad k = 1, \ldots, v$$

- **pour** $i = 1, \ldots, N$

$$Q_k \rightarrow Q_k \alpha^k \quad \text{avec} \quad k = 1, \ldots, v$$

$$\text{Si} \quad \sum_{k=1}^{v} Q_k = 1 \quad \text{alors il y a une erreur en position} \quad N - i$$

Note : la dernière ligne s'explique par la relation $\sum_{k=1}^{v} Q_k = \sigma(\alpha^i) + 1$.

Exemple (suite) :

On peut vérifier que

$$\begin{aligned} \sigma(p) = \sigma^{(4)}(p) &= 1 + \alpha^{14}p + \alpha^{13}p^2 \\ &= (1+\alpha^4 p)(1+\alpha^9 p) \end{aligned} \tag{9.38}$$

Les racines de $\sigma(p)$ sont égales à α^{11} et α^6. Les inverses de ces racines sont $\beta_1 = \alpha^{15-11} = \alpha^4$ et $\beta_2 = \alpha^{15-6} = \alpha^9$. On en déduit donc bien que les erreurs sont aux positions $j_1 = 4$ et $j_2 = 9$.

Nous pouvons également appliquer la méthode de Chien pour déterminer la position des erreurs. Les résultats sont présentés dans le tableau suivant :

On retrouve bien que les deux erreurs sont aux positions $j_1 = 4$ et $j_2 = 9$.

TAB. 9.14 – détermination de la position des erreurs par la méthode de Chien

i	Q_1	Q_2	$\sum_k Q_k$
0	α^{14}	α^{13}	α^2
1	1	1	0
2	α	α^2	α^5
3	α^2	α^4	α^{10}
4	α^3	α^6	α^2
5	α^4	α^8	α^5
6	α^5	α^{10}	$1 \quad \rightarrow j_1 = 15 - 6 = 9$
7	α^6	α^{12}	α^4
8	α^7	α^{14}	α
9	α^8	α	α^{10}
10	α^9	α^3	α
11	α^{10}	α^5	$1 \quad \rightarrow j_2 = 15-11 = 4$

9.7 Les codes Reed-Solomon

Les codes Reed-Solomon (RS) sont construits sur le corps fini GF(2^m) où m est un entier. Les $2^m - 1$ éléments non nuls du corps GF(2^m) sont les puissances successives de l'élément primitif α racine de $\alpha^{2^m-1} - 1 = 0$. Chaque polynôme minimal associé à un élément non nul $\alpha^i (i = 0, 1, \ldots, 2^m - 2)$ est de la forme $m_i(p) = p - \alpha^i$.

Le polynôme générateur d'un code RS(N, K) de longueur N et de dimension K est le produit des $N - K$ polynômes minimaux $m_i(p)$ avec $i = 1, 2, ..., N - K$:

$$\begin{aligned} g(p) &= \sum_{i=1}^{N-K} m_i(p) \\ &= \sum_{i=1}^{N-K} (p - \alpha^i) \end{aligned} \tag{9.39}$$

La distance minimale de Hamming d_{min} de ces codes est égale à $N-K+1$. Ces codes atteignent donc la borne de Singleton.

Par rapport aux codes binaires, la capacité de correction des codes RS est donc de $e = \frac{N-K}{2}$ symboles de m bits soit au maximum me bits. Par exemple, un code RS$(255, 239, 17)$ dans GF(2^8)peut corriger jusqu'à 8 octets mais ne pourra corriger 64 bits que si ces bits erronés sont localisés dans seulement 8 octets.

Le taux d'erreurs mot est égal à :

$$TEM = \sum_{i=e+1}^{N} C_N^i (TES_E)^i (1 - TES_E)^{N-i}$$

Par conséquent, le taux d'erreurs symbole TES_S en sortie d'un décodeur de Reed-Solomon à entrées dures est le suivant :

$$TES_S = \frac{1}{N} \sum_{i=e+1}^{N} i C_N^i (TES_E)^i (1 - TES_E)^{N-i}$$

où TES_E est le taux d'erreurs symbole en entrée.

La courbe de performance $TES_S = f(TES_E)$ pour le code Reed Solomon $(255, 239, 17)$ est présentée sur la figure 9.7.

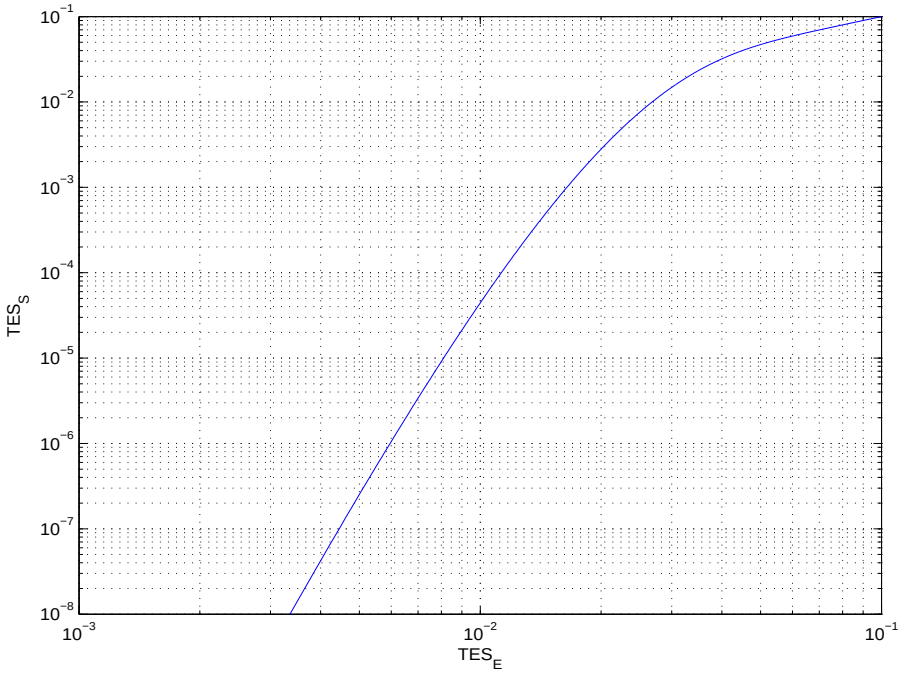


FIG. 9.8 – $TES_S = f(TES_E)$ pour le code Reed Solomon $(255, 239, 17)$.

D'un point de vue décodage, ces codes sont décodés comme les codes BCH. On doit seulement ajouter une opération qui consiste à évaluer l'amplitude des erreurs. Cette évaluation permet de déterminer la postion des bits erronés à l'intérieur d'un symbole erroné.

Les codes RS sont particulièrement bien adaptés à la correction de paquets d'erreurs dans les systèmes de communication.

Chapitre 10

Codes convolutifs

10.1 Introduction

Dans ce chapitre, nous présenterons les codes convolutifs binaires et leurs codeurs. Puis, nous introduirons l'algorithme de Viterbi utilisé pour le décodage de ces codes.

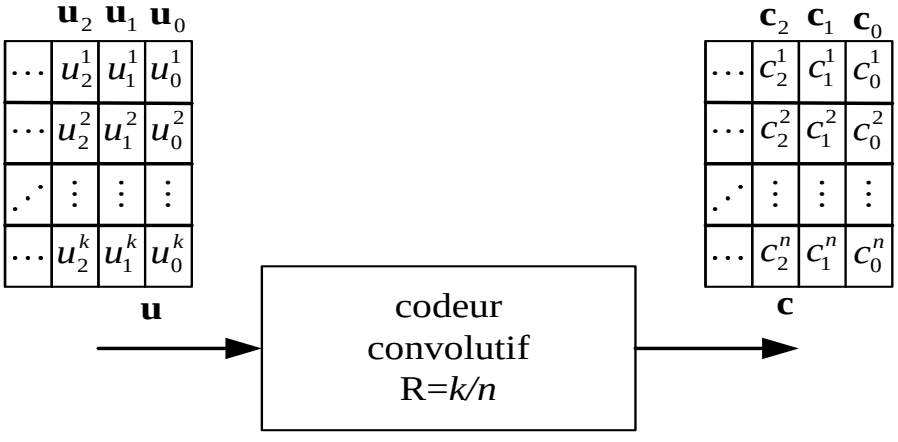
Pour plus d'informations sur les codes convolutifs nous renvoyons le lecteur à l'article de référence de Forney [11] et aux livres de Viterbi et Omura [30] et de Johannesson et Zigangirov [17]. Nous nous limiterons ici au cas des codes convolutifs binaires.

10.2 Les codes convolutifs

10.2.1 Structures et représentations mathématiques

Un code convolutif est un code qui transforme une séquence semi-infinie de mots d'informations en une autre séquence semi-infinie de mots de code.

Soit $\mathbf{u}$ la séquence d'entrée ou séquence de mots d'information de dimension k d'un codeur convolutif binaire de rendement $\frac{k}{n}$, $\mathbf{u} = \mathbf{u}_0, \mathbf{u}_1, \mathbf{u}_2, \ldots$ avec $\mathbf{u}_i = [u_i^1, u_i^2, ..., u_i^k]$ et $\mathbf{c}$ la séquence de sortie ou séquence de mots de code de dimension n, $\mathbf{c} = \mathbf{c}_0, \mathbf{c}_1, \mathbf{c}_2, \ldots$ avec $\mathbf{c}_i = [c_i^1, c_i^2, ..., c_i^n]$.



Il est souvent plus simple d'utiliser l'opérateur de délai D pour décrire ces séquences (l'opéra-

teur D doit être ici considéré comme un marqueur de position). On a alors :

$$\mathbf{c}(D) = [c^1(D), c^2(D), ..., c^n(D)] \qquad \text{et} \qquad \mathbf{u}(D) = [u^1(D), u^2(D), ..., u^k(D)] \tag{10.1}$$

avec

$$u^j(D) = \sum_{i=0}^{\infty} u_i^j D^i \qquad \text{et} \qquad c^j(D) = \sum_{i=0}^{\infty} c_i^j D^i \qquad \text{avec} \qquad u_i^j, c_i^j \in \mathbb{F}_2 \tag{10.2}$$

où $\mathbb{F}_2$ est le corps de Galois à 2 éléments. L'ensemble des séries de Laurent causales de la forme

$$a(D) = \sum_{i=0}^{\infty} a_i D^i \qquad \text{avec} \qquad a_i \in \mathbb{F}_2$$

constitue un idéal noté $\mathbb{F}_2[[D]]$, sous ensemble du corps des séries de Laurent $\mathbb{F}_2((D))$. Il faut noter que certains auteurs préfèrent considérer les séquences $\mathbf{u}(D)$ et $\mathbf{c}(D)$ comme des séries de Laurent.

Définition 1 : un codeur convolutif binaire de rendement k/n est la réalisation par un circuit linéaire de l'association ϕ d'une séquence de mots d'information de k bits $\mathbf{u}(D)$ avec une séquence de mots de code de n bits $\mathbf{c}(D)$:

$$\begin{aligned} \phi : &\mathbb{F}_2^k[[D]] \rightarrow \mathbb{F}_2^n[[D]] \\ &\mathbf{u}(D) \rightarrow \mathbf{c}(D) \end{aligned}$$

avec

$$\mathbf{c}(D) = \mathbf{u}(D)\mathbf{G}(D) \tag{10.3}$$

$\mathbf{G}(D)$ est la matrice génératrice utilisée par le codeur. $\mathbf{G}(D)$ est de dimension $k \times n$ et de rang k.

$$\mathbf{G}(D) = \begin{pmatrix} g_{1,1}(D) & g_{1,2}(D) & \cdots & g_{1,n}(D) \\ g_{2,1}(D) & g_{2,2}(D) & \cdots & g_{2,n}(D) \\ \cdots & \cdots & \cdots & \cdots \\ g_{k,1}(D) & g_{k,2}(D) & \cdots & g_{k,n}(D) \end{pmatrix} \tag{10.4}$$

Les éléments $g_{i,j}(D)$ de la matrice génératrice $\mathbf{G}(D)$ sont des polynômes de D ou des fonctions rationnelles de polynômes de D.

Pour une matrice génératrice fixée, plusieurs architectures de codeurs sont possibles. Il existe 2 familles de codeurs : les codeurs non récursifs et les codeurs récursifs.

Dans le cas d'un codeur non récursif, les n bits de sortie dépendent des k bits d'entrée courants et d'un nombre fini de bits d'entrée précédents. Ce codeur peut être vu comme un système à réponse impulsionnelle finie (RIF) sur le corps de Galois $\mathbb{F}_2$. Tous les éléments $g_{i,j}(D)$ de la matrice génératrice sont alors des polynômes dans D.

Définition 2 : un code convolutif est l'ensemble de toutes les séquences de sortie possibles $\mathbf{c}(D)$ du codeur convolutif.

Pour un code convolutif, il existe un ensemble de matrices génératrices $\mathbf{G}(D)$ et de codeurs qui génèrent celui-ci.

Définition 3 : deux matrices génératrices $G(D)$ et $G'(D)$ sont équivalentes si elles engendrent le même code. Deux codeurs convolutifs sont équivalents si leurs matrices génératrices sont équivalentes.

Pour un codeur donné, soit M_i le nombre de cellules mémoires associées à la ième séquence d'entrée binaire :

$$M_i = \max_{i \leq j \leq n} \deg g_{ij}(D)$$

Le nombre total de cellules mémoires est égal à $M = \sum_{i=1}^{k} M_i$. M détermine la complexité du codeur.

Exemple 1 : Considérons l'exemple du codeur convolutif non récursif $k = 1$, $n = 2$ $M = 2$ donné sur la figure 10.2.1.

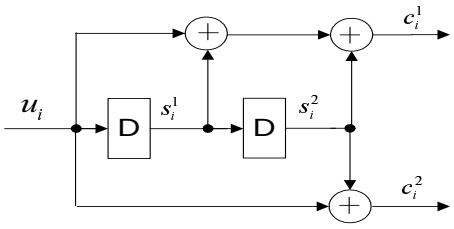


FIG. 10.1 – codeur convolutif non récursif de rendement 1/2 M=2 g_1=7, g_2=5.

Sa matrice génératrice est la suivante :

$$\mathbf{G}(D) = (g_1(D), g_2(D)) = (1 + D + D^2, 1 + D^2)$$

Le degré maximum des polynômes étant égal à 2, la réalisation de ce codeur nécessitera $M = 2$ cellules mémoire.

On décrit également les polynômes générateurs du codeur convolutif sous une forme octale :

$$g_1(D) = g_0^1 + g_1^1 D + g_2^1 D^2 = 1 + D + D^2$$

$$g_1 = [111]_{\text{bin}} = 7_{\text{oct}}$$

$$g_2(D) = g_0^2 + g_1^2 D + g_2^2 D^2 = 1 + D^2$$

$$g_2 = [101]_{\text{bin}} = 5_{\text{oct}}$$

Dans cet exemple on a les relations :

$$u(D) = u_0 + u_1 D + u_2 D^2 + \dots$$

$$c^1(D) = c_0^1 + c_1^1 D + c_2^1 D^2 + \cdots = u(D)(1 + D + D^2) \tag{10.5}$$

$$c^2(D) = c_0^2 + c_1^2 D + c_2^2 D^2 + \cdots = u(D)(1 + D^2) \tag{10.6}$$

Les équations 10.2.1 et 10.2.1 peuvent aussi à l'instant i s'écrire :

$$c_i^1 = g_0^1 u_i + g_1^1 u_{i-1} + g_2^1 u_{i-2} = u_i + u_{i-1} + u_{i-2}$$

$$c_i^2 = g_0^2 u_i + g_1^2 u_{i-1} + g_2^2 u_{i-2} = u_i + u_{i-2}$$

Le rendement de ce code est égal à 1/2.

Exemple 2 : Considérons l'exemple du codeur convolutif non récursif $k = 1$, $n = 2$ $M = 6$ donné sur la figure 10.2.1.

Sa matrice génératrice est la suivante :

$$\mathbf{G}(D) = (g_1(D), g_2(D)) = (1 + D^2 + D^3 + D^5 + D^6, 1 + D + D^2 + D^3 + D^6)$$

Ce codeur est un standard utilisé dans le domaine spatial et les télécommunications.

Pour le codeur récursif, les n bits de sortie peuvent dépendre d'un nombre infini de bits d'entrée précédents. C'est pourquoi le codeur récursif est un système à réponse impulsionnelle infinie (RII) sur $\mathbb{F}_2$.

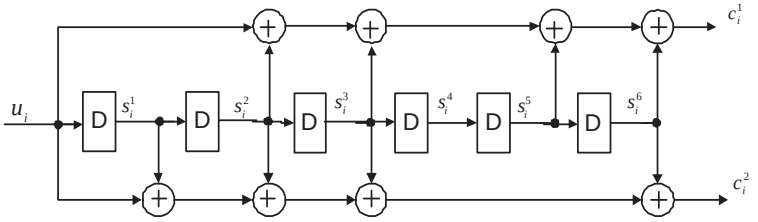


FIG. 10.2 – codeur convolutif non récursif de rendement 1/2 M=6 g_1=133, g_2=171.

Pour les codeurs récursifs, les éléments $g_{i,j}(D)$ de la matrice génératrice sont des rapports de polynômes.

Un codeur convolutif binaire de rendement k/n est appelé systématique si les k bits de l'entrée sont recopiés à la sortie du codeur. Sa matrice génératrice s'écrit alors :

$$\mathbf{G}(D) = (I_k \quad R(D))$$

Exemple 3 Considérons l'exemple du codeur convolutif récursif systématique $k = 1$, $n = 2$ $M = 2$ donné sur la figure 10.2.1.

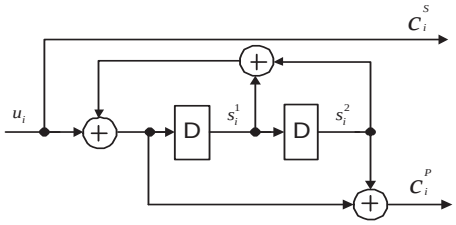


FIG. 10.3 – codeur convolutif récursif systématique de rendement 1/2 M=2.

La matrice génératrice de ce codeur convolutif récursif systématique est la suivante :

$$\mathbf{G}(D) = \left[1, \frac{1+D^2}{1+D+D^2}\right]$$

Les codeurs de l'exemple 1 et 3 sont équivalents.

Les codeurs convolutifs récursifs ont été peu utilisés car ceux-ci produisent les mêmes codes convolutifs que les codeurs convolutifs non récursifs. Cependant, les travaux de Battail [**?**] ont montré qu'un codeur convolutif récursif imitait le codeur aléatoire si le polynôme dénominateur était primitif. Ceci justifie leur utilisation dans les codes convolutifs concaténés en parallèle.

10.2.2 Représentation matricielle des codes convolutifs

Par rapport au codeur en bloc, un codeur convolutif contient des cellules mémoires internes stockant un vecteur d'états composé de M bits. Ainsi, le mot de code $\mathbf{c}_i$ est une fonction linéaire du mot d'information $\mathbf{u}_i$ mais aussi de l'état interne du codeur à l'instant i, $\mathbf{s}_i$.

Les équations reliant $\mathbf{u}_i$, $\mathbf{s}_i$, $\mathbf{s}_{i+1}$ et $\mathbf{c}_i$ sont les suivantes :

$$\begin{cases} \mathbf{s}_{i+1} = \mathbf{s}_i\mathcal{A} + \mathbf{u}_i\mathcal{B} \\ \mathbf{c}_i = \mathbf{s}_i\mathcal{C} + \mathbf{u}_i\mathcal{D} \end{cases} \tag{10.7}$$

Alors qu'un codeur en bloc linéaire binaire (N, K) est défini par une seule matrice $\mathbf{G}$ de dimension $K \times N$, un codeur convolutif de rendement k/n est défini par les 4 matrices $\mathcal{A}, \mathcal{B}, \mathcal{C}$ et

$\mathcal{D}$ dont les dimensions sont les suivantes :

$$\begin{aligned}\mathcal{A} &: M \times M \\ \mathcal{B} &: k \times M \\ \mathcal{C} &: M \times n \\ \mathcal{D} &: k \times n\end{aligned} \tag{10.8}$$

Ces équations correspondent exactement aux équations d'état et de sortie utilisées pour décrire un système linéaire discret à entrées et sorties multiples et invariant dans le temps dans la théorie des systèmes.

Pour l'exemple 1 ci-dessus, on a également les relations suivantes :

$$s_{i+1}^1 = u_i \tag{10.9}$$

$$s_{i+1}^2 = s_i^1 \tag{10.10}$$

$$c_i^1 = u_i + s_i^1 + s_i^2 \tag{10.11}$$

$$c_i^2 = u_i + s_i^2 \tag{10.12}$$

Ainsi, nous avons les équations d'état et de sortie suivantes :

$$\mathbf{s}_{i+1} = \begin{pmatrix} s_{i+1}^1 & s_{i+1}^2 \end{pmatrix} = \begin{pmatrix} s_i^1 & s_i^2 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} + u_i \begin{pmatrix} 1 & 0 \end{pmatrix} \tag{10.13}$$

$$\mathbf{c}_i = \begin{pmatrix} c_i^1 & c_i^2 \end{pmatrix} = \begin{pmatrix} s_i^1 & s_i^2 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} + u_i \begin{pmatrix} 1 & 1 \end{pmatrix} \tag{10.14}$$

A partir des équations d'état et de sortie (10.2.2), nous avons les relations suivantes en utilisant l'opérateur de délai unité D :

$$\begin{cases} \mathbf{s}(D)D^{-1} = \mathbf{s}(D)\mathcal{A} + \mathbf{u}(D)\mathcal{B} \\ \mathbf{c}(D) = \mathbf{s}(D)\mathcal{C} + \mathbf{u}(D)\mathcal{D} \end{cases} \tag{10.15}$$

où $\mathbf{u}(D)$ et $\mathbf{c}(D)$ sont définis dans (10.2.1) et (10.2.1) et

$$\mathbf{s}(D) = [s^1(D), s^2(D), ..., s^M(D)] \qquad \text{avec} \qquad s^j(D) = \sum_{i=0}^{\infty} s_i^j D^i$$

On obtient alors la relation classique entre la matrice génératrice $\mathbf{G}(D)$ et les matrices de la réalisation $\mathcal{A}$,$\mathcal{B}$,$\mathcal{C}$ et $\mathcal{D}$.

$$\mathbf{G}(D) = \mathcal{D} + \mathcal{B}(D^{-1}\mathcal{I}_M - \mathcal{A})^{-1}\mathcal{C} \tag{10.16}$$

Une dernière représentation matricielle des codes convolutifs s'obtient en sérialisant les bits d'entrée et de sortie du codeur convolutif : Soit

$$\mathbf{u} = u_0^1, u_0^2, \ldots, u_0^k, u_1^1, u_1^2, ..., u_1^k, \ldots$$

$$\mathbf{c} = c_0^1, c_0^2, \ldots, c_0^n, c_1^1, c_1^2, ..., c_1^n, \ldots$$

La relation (10.2.1) devient alors :

$$\mathbf{c} = \mathbf{uG} \tag{10.17}$$

où $\mathbf{G}$ est une matrice génératrice semi-infinie dont les éléments sont binaires.

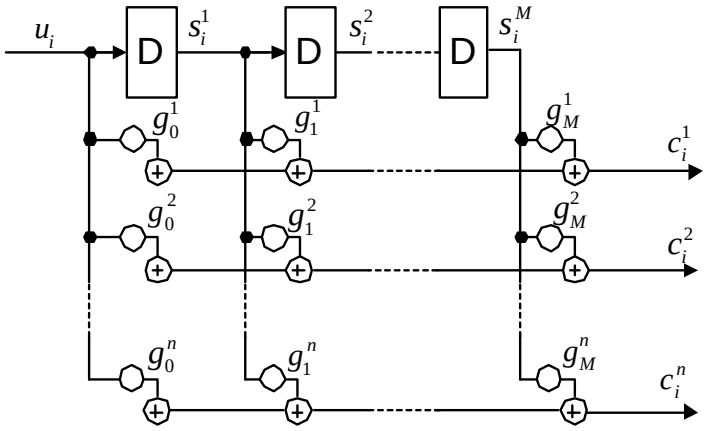


FIG. 10.4 – codeur convolutif non récursif générique de rendement $1/n$.

Nous ne décrirons la matrice $\mathbf{G}$ que dans le cas des codeurs convolutifs avec $k = 1$ (rendement $1/n$) de la figure 10.2.2. Le cas général se déduit sans aucune difficulté.

La matrice $\mathbf{G}$ est alors de la forme :

$$\mathbf{G} = \begin{pmatrix} \mathbf{g}_0 & \mathbf{g}_1 & \mathbf{g}_2 & \cdots & \mathbf{g}_M & & & \\ & \mathbf{g}_0 & \mathbf{g}_1 & \cdots & \mathbf{g}_{M-1} & \mathbf{g}_M & & \\ & & \mathbf{g}_0 & \cdots & \mathbf{g}_{M-2} & \mathbf{g}_{M-1} & \mathbf{g}_M & \\ & & & \ddots & \ddots & \ddots & \ddots & \ddots \end{pmatrix} \tag{10.18}$$

avec $\mathbf{g}_i = [g_i^1 g_i^2 \dots g_i^n]$. Les espaces blancs correspondent à des zéros

Pour l'exemple 1, la matrice génératrice est la suivante :

$$\mathbf{G} = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & & & & \\ & & 1 & 1 & 1 & 0 & 1 & & \\ & & & & 1 & 1 & 1 & 0 & 1 \\ & & & & & & \ddots & & \end{pmatrix} \tag{10.19}$$

Comme pour les codes en bloc, on peut introduire la matrice de parité $\mathbf{H}$ semi-infinie suivante :

$$\mathbf{G}\mathbf{H}^T = \mathbf{0} \tag{10.20}$$

Chaque séquence $\mathbf{c}$ devra donc satisfaire la relation

$$\mathbf{c}\mathbf{H}^T = \mathbf{0} \tag{10.21}$$

Pour l'exemple 1, la matrice de parité est la suivante :

$$\mathbf{H} = \begin{pmatrix} 1 & 1 & & & & & & & & \\ 0 & 1 & 1 & 1 & & & & & & \\ 1 & 1 & 0 & 1 & 1 & 1 & & & & \\ & & 1 & 1 & 0 & 1 & 1 & 1 & & \\ & & & & 1 & 1 & 0 & 1 & 1 & 1 \\ & & & & & & & & & \ddots \end{pmatrix} \tag{10.22}$$

10.3 Représentations graphiques des codes convolutifs

10.3.1 Diagramme de transitions d'état

Le diagramme de transitions d'état permet de décrire simplement le fonctionnement du codeur convolutif. Pour un codeur convolutif récursif composé de M cellules mémoires, on définit l'état interne du codeur à l'instant i par un vecteur $\mathbf{s}_i$ de dimension M :

$$\mathbf{s}_i = [s_i^1, s_i^2, ...s_i^M] \ .$$

s_i^j est l'état à l'instant i de la j-ième cellule mémoire.

Le diagramme de transition d'état est un graphe orienté composé de 2^M sommets. Chaque sommet correspond à un état interne du codeur.

Un exemple de diagramme de transition pour le codeur convolutif non récursif de rendement $1/2$ de l'exemple 1 défini par la matrice génératrice $G(D) = (1 + D + D^2, 1 + D^2)$ est donné figure 10.5.

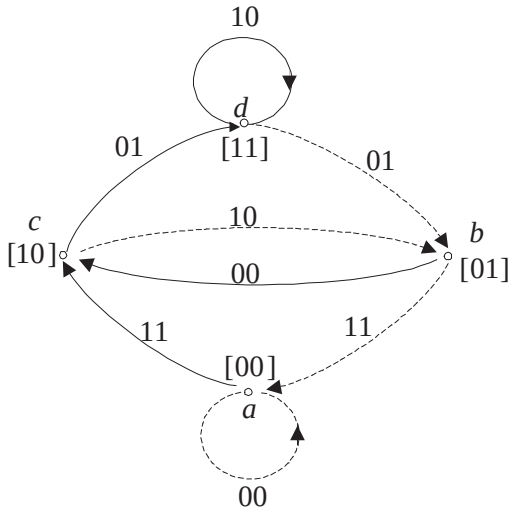


FIG. 10.5 – Diagramme d'état pour le code convolutif non récursif $g_1 = 7, g_2 = 5$.

Dans cet exemple on a $M = 2$ et donc $2^M = 4$ états internes du codeur :

TAB. 10.1 – Description des états internes

état interne	s_i^1	s_i^2
a	0	0
b	0	1
c	1	0
d	1	1

Chaque branche est renseignée avec les bits de sortie (ici c_i^1 et c_i^2). Les branches en traits pointillés et continus correspondent respectivement à un bit d'entrée égal à 0 et à 1.

10.3.2 Diagramme en treillis

A partir du diagramme de transitions d'état, il est possible de construire un graphe biparti appelé aussi treillis élémentaire. Chaque branche b de ce graphe biparti relie un état de départ

$s^-(b)$ et un état d'arrivée $s^+(b)$. Le treillis élémentaire pour le codeur convolutif non récursif de l'exemple 1 est donné figure 10.6.

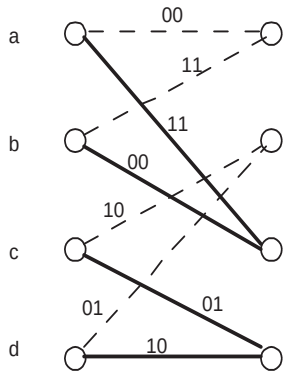


FIG. 10.6 – Treillis élémentaire d'un codeur convolutif non récursif $g_1 = 7, g_2 = 5$.

A chaque noeud arrivent et partent 2^k branches. Le nombre total de branche est donc égal à 2^{k+M} ; (dans cet exemple $k = 1$, $M = 2$ et donc $2^{k+M} = 8$).

Le diagramme en treillis est un diagramme de transitions d'état où l'abscisse correspond au temps. Il est obtenu simplement en assemblant un nombre infini de treillis élémentaires. Le diagramme en treillis pour le codeur convolutif non récursif de rendement $1/2$ de l'exemple 1 est donné figure 10.7.

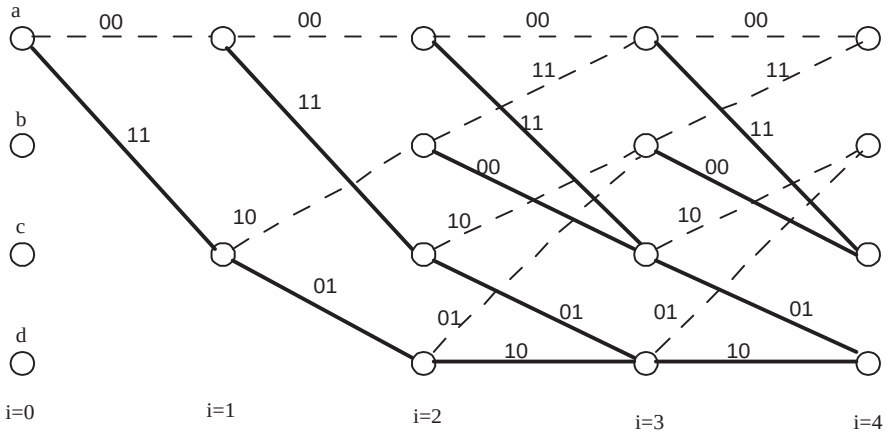


FIG. 10.7 – Diagramme en treillis d'un codeur convolutif non récursif $g_1 = 7, g_2 = 5$.

Sur chaque branche on renseigne la valeur des deux bits de sortie c_i^1 et c_i^2. Dans ce diagramme en treillis, les traits gras correspondent à $u_i = 1$ alors que les traits pointillés correspondent à $u_i = 0$. On peut constater sur ce diagramme qu'après une période transitoire, le diagramme en treillis est bien la concaténation de treillis élémentaires.

10.3.3 Graphes TWL

Les graphes TWL (Tanner, Wiberg et Loeliger) [**?**] sont utilisés pour décrire à la fois le codage et le décodage des codes convolutifs et des codes convolutifs concaténés.

En plus des nœuds de variable binaire et des nœuds de contrôle de parité, nous utiliserons une troisième famille de nœuds appelés nœuds de variable d'état ou nœuds de variable binaire cachée . Nous représenterons les nœuds de variable cachée par un double cercle. De plus, les nœuds de contrôle de parité peuvent être remplacés par des nœuds de contrôle plus généraux appelés fonctions locales. Ces fonctions locales sont symbolisées par des carrés noirs dans ces graphes.

Le graphe TWL d'un codeur convolutif systématique de rendement 1/2 est donné figure 10.8.

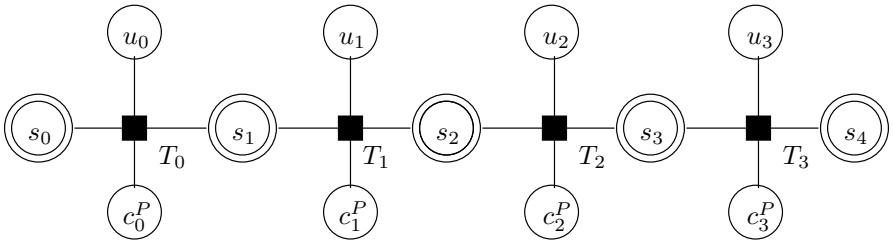


FIG. 10.8 – Graphe TWL d'un codeur convolutif systématique de rendement 1/2.

La fonction locale $T_i(s_i, c_i^P, u_i, s_{i+1})$ à la position i est définie à partir des équations d'état et de sortie (10.2.2) du codeur convolutif. On a :

$$T_i(s_i, c_i^P, u_i, s_{i+1}) = \begin{cases} 1 & \text{si les équations (10.2.2) sont valides} \\ 0 & \text{sinon} \end{cases} \tag{10.23}$$

La fonction locale $T_i(s_i, c_i^P, u_i, s_{i+1})$ correspond au treillis élémentaire du codeur convolutif qui relie s_i, s_{i+1}, c_i^P et u_i.

10.4 Distance minimale et fonction de transfert des codes convolutifs

La détermination de la distance minimale d'un code convolutif peut s'obtenir à partir du diagramme en treillis. Pour déterminer la distance minimale on prend comme chemin de référence le chemin associé à la séquence d'entrée nulle $\mathbf{u} = [0, 0, \ldots]$. Ce chemin est représenté en gras sur le diagramme en treillis de la figure 10.9. Sur chaque branche, on a noté le poids de Hamming des deux bits de sortie c_i^1 et c_i^2.

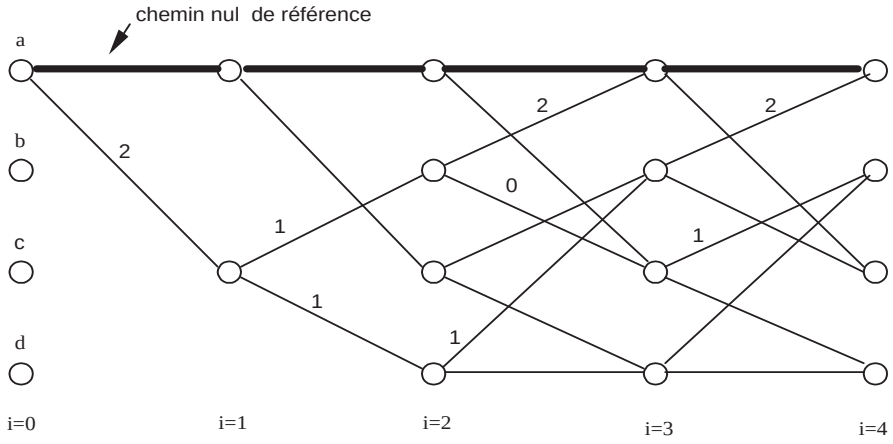


FIG. 10.9 – Diagramme en treillis d'un codeur convolutif non récursif $g_1 = 7, g_2 = 5$.

A partir de ce diagramme on constate qu'un seul chemin quittant le chemin de référence à l'instant $i = 0$ est à la distance 5 du chemin de référence. Il s'agit du chemin (a, c, b, a). Ainsi, la distance minimale de ce code convolutif est égale à 5. Il est à noter qu'il existe deux chemins (a, c, d, b, a) et (a, c, b, c, b, a) à la distance 6 du chemin de référence.

La fonction de transfert d'un code convolutif permet de déterminer les propriétés de distance de celui-ci. Cette fonction s'obtient à partir du diagramme d'états du codeur convolutif. Nous allons montrer comment calculer cette fonction de transfert en utilisant l'exemple 1. Le diagramme d'état de ce codeur convolutif a été donné sur la figure 10.5.

Pour déterminer la fonction de transfert, nous devons modifier ce diagramme comme suit : tout d'abord chaque branche est renseignée avec $D^{w_O} N^{w_I}$ où w_I et w_O sont respectivement le poids de Hamming des bits d'information et des bits de sortie. Dans notre exemple, un bit d'information et deux bits de sorties sont associés à chaque branche. Comme précédemment, les branches en traits pointillés et continus correspondent respectivement à un bit d'entrée égal à 0 et à 1. Ensuite, nous supprimons la branche se rebouclant sur l'état a qui n'intervient pas sur les propriétés de distance du code. Finalement, nous séparons le noeud a en deux noeuds distincts a et e correspondant respectivement à au noeud d'entrée et au noeud de sortie du diagramme d'état modifié. Le diagramme d'état modifié pour l'exemple 1 est donné sur la figure 10.10.

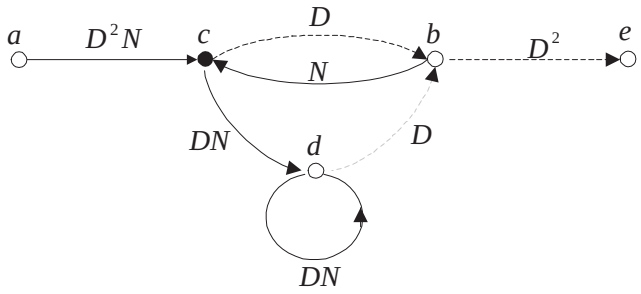


FIG. 10.10 – Diagramme d'états modifié d'un codeur convolutif non récursif $g_1 = 7, g_2 = 5$.

A partir de ce diagramme, nous pouvons alors calculer la fonction de transfert du code convolutif. Nous avons les relations suivantes :

Nous avons les 4 équations suivantes :

$$\begin{aligned} X_c &= ND^2X_a + NX_b \\ X_b &= DX_c + DX_d \\ X_d &= NDX_c + NDX_d \\ X_e &= D^2X_b \end{aligned}$$

En résolvant le système d'équations ci-dessus, on obtient finalement la fonction de transfert suivante :

$$T(D, N) = \frac{X_e}{X_a} = \frac{ND^5}{1 - 2ND} = ND^5 + 2N^2D^6 + 4N^3D^7 + 8N^4D^8 + 16N^5D^9 + \dots$$

Cette fonction nous renseigne en particulier sur la distance minimale du code (ici 5) et le nombre de séquences associées (ici une seule séquence). La fonction de transfert permet de prédire les performances du code convolutif.

Sur la table suivante nous présentons les meilleurs codes convolutifs de rendement $1/2$:

nbr mémoire	code	d_{min}
1	$(2,3)$	3
2	$(5,7)$	5
3	$(15,17)$	6
4	$(23,35)$	7
5	$(53,75)$	8
6	$(133,171)$	10

10.5 Algorithme de Viterbi

Soit la séquence $\mathbf{x} = (x_0, x_1, x_2, \ldots, x_{N-1})$ de longueur N envoyée dans un canal discret stationnaire sans mémoire de densité de probabilité conditionnelle $p(y/x)$ et $\mathbf{y} = (y_0, y_1, y_2, \ldots, y_{N-1})$ la séquence reçue.

Nous avons vu précédemment qu'un décodeur à *maximum de vraisemblance* (*maximum likelihood* en anglais ou ML) recherche la séquence $\hat{\mathbf{x}}$ pour laquelle la probabilité conditionnelle $Pr(\mathbf{y}|\mathbf{x})$ est la plus grande.

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} Pr(\mathbf{y}|\mathbf{x}) \tag{10.24}$$

En considérant que les signaux émis sont perturbés indépendamment les uns des autres, la probabilité conditionnelle est égale au produit des probabilités conditionnelles $Pr(y_i|x_i)$:

$$Pr(\mathbf{y}|\mathbf{x}) = \prod_{i=0}^{N-1} Pr(y_i|x_i) \tag{10.25}$$

L'algorithme de Viterbi [29][12] appliqué sur le treillis du code convolutif est l'algorithme ML généralement utilisé pour le décodage des codes convolutifs. D'autres décodages comme le décodage séquentiel peuvent également être utilisés lorsque le nombre de mémoires est élevé ($M > 10$) [17].

La recherche de la séquence $\hat{\mathbf{x}}$ la plus probable est équivalente à déterminer le chemin ou la séquence d'état le plus probable. La séquence $\hat{\mathbf{x}}$ se déduit alors immédiatement.

Pour le canal binaire symétrique de probabilité d'inversion p, la probabilité $Pr(\mathbf{y}|\mathbf{x})$ est la suivante :

$$Pr(\mathbf{y}|\mathbf{x}) = p^{d_H(\mathbf{y},\mathbf{x})}(1-p)^{N-d_H(\mathbf{y},\mathbf{x})} = (1-p)^N \left(\frac{p}{1-p}\right)^{d_H(\mathbf{y},\mathbf{x})} \tag{10.26}$$

où $d_H(\mathbf{y},\mathbf{x})$ est la distance de Hamming entre la séquence reçue $\mathbf{y}$ et la séquence $\mathbf{x}$. Comme p est compris entre 0 et 0.5, on a $0 < \frac{p}{1-p} < 1$. Nous avons démontré que maximiser $Pr(\mathbf{y}|\mathbf{x})$ revient à minimiser la distance de Hamming entre $\mathbf{y}$ et $\mathbf{x}$.

Considérons maintenant le cas du décodage à entrées souples d'une séquence reçue après filtrage adapté pour un canal BBAG. Nous avons vu que nous pouvons exprimer à l'instant i sa sortie y_i comme suit :

$$y_i = x_i + n_i \tag{10.27}$$

avec E_s énergie moyenne par symbole x_i et n_i échantillon bruit blanc gaussien centré de variance $\sigma^2 = \frac{N_0}{2}$. Ainsi la sortie du démodulateur la densité de probabilité de y_i conditionnellement à x_i est :

$$p(y_i|x_i) = \frac{1}{\sqrt{\pi N_0}} \exp\left\{ -\frac{[y_i - x_i]^2}{N_0} \right\} \tag{10.28}$$

Comme la fonction logarithme est croissante, au lieu d'utiliser la relation (10.5) pour déterminer $\hat{\mathbf{x}}$, on peut utiliser :

$$\begin{aligned}\hat{\mathbf{x}} &= \arg\max_{\mathbf{x}} \ln Pr(\mathbf{y}|\mathbf{x}) \\ &= \arg\max_{\mathbf{x}} \ln \prod_{i=0}^{N-1} Pr(y_i|x_i) \qquad (10.29) \\ &= \arg\max_{\mathbf{x}} \sum_{i=0}^{N-1} \ln Pr(y_i|x_i) \qquad (10.30)\end{aligned}$$

On obtient donc une première version du décodeur à maximum de vraisemblance :

$$\begin{aligned}\hat{\mathbf{x}} &= \arg\max_{\mathbf{x}} \ln Pr(\mathbf{y}|\mathbf{x}) \\ &= \arg\max_{\mathbf{x}} \sum_{i=0}^{N-1} \left\{ -\frac{[y_i - x_i]^2}{N_0} \right\} \\ &= \arg\min_{\mathbf{x}} \sum_{i=0}^{N-1} (y_i - x_i)^2 \qquad (10.31)\end{aligned}$$

Une seconde version du décodeur à maximum de vraisemblance est obtenue en approximant les calculs comme suit :

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \sum_{i=0}^{N-1} |y_i - x_i| \qquad (10.32)$$

Cette version sous optimale donne cependant de bonnes performances et est souvent utilisée en pratique.

Dans le cas d'une modulation bipodale ($x_i = \pm\sqrt{E_s}$) il est possible d'obtenir une troisième version exacte :

$$\begin{aligned}\hat{\mathbf{x}} &= \arg\max_{\mathbf{x}} \ln Pr(\mathbf{y}|\mathbf{x}) \\ &= \arg\max_{\mathbf{x}} \sum_{i=0}^{N-1} -(y_i - x_i)^2 \\ &= \arg\max_{\mathbf{x}} \sum_{i=0}^{N-1} -y_i^2 + 2y_i x_i - x_i^2 \\ &= \arg\max_{\mathbf{x}} \sum_{i=0}^{N-1} y_i x_i \qquad (10.33)\end{aligned}$$

En effet, comme y_i^2, et $x_i^2 = E_s$ sont communs à toutes les séquences on peut simplifier sensiblement les calculs sans aucune dégradation des performances.

L'algorithme de Viterbi utilisé pour le décodage à entrées dures ou souples permet de déterminer la séquence $\hat{\mathbf{x}}$ en évitant de calculer les métriques cumulées associées à chacune des séquences possibles.

Le principe général de l'algorithme de Viterbi consiste à chaque section du diagramme en treillis à éliminer tous les chemins (et les séquences associées) qui ne peuvent pas être le chemin le plus vraisemblable. A chaque noeud du treillis, on ne conserve qu'un seul chemin

Pour comprendre l'algorithme de Viterbi nous allons utiliser le codeur de l'exemple 1. Le décodeur sera ici à entrées dures ; on utilisera donc comme métrique la distance de Hamming.

Supposons qu'à l'instant $i = 0$, le codeur convolutif soit dans l'état a et que la séquence en entrée du codeur soit 1001. La séquence en sortie est alors 11 10 11 11. On considère que la séquence reçue est 11 00 11 11 (soit une erreur sur le 3ième bit).

A chaque étape l'algorithme de Viterbi calcule pour chaque nouvelle branche la distance de Hamming entre le couple de bits reçus et le couple de bits associés à la branche considérée. Puis il calcule les 2^{k+M} métriques cumulées.

Le diagramme en treillis associé est présenté sur la figure 10.11.

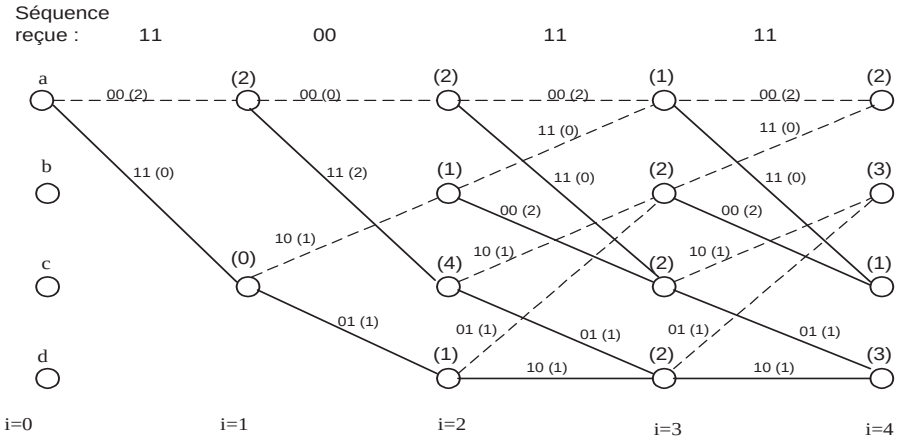


FIG. 10.11 – Diagramme en treillis d'un codeur convolutif non récursif $g_1 = 7, g_2 = 5$.

Ensuite pour chaque noeud, on ne conserve qu'un seul chemin baptisé chemin survivant : le chemin survivant étant le chemin qui est à la distance de Hamming minimale de la séquence reçue.

Exemple : à l'instant i=3, deux chemins convergent vers l'état a :
- le chemin (a, c, b, a) à la distance 0+1+0=1 de la séquence reçue.
- le chemin (a, a, a, a) à la distance 2+0+2=4 de la séquence reçue.

Le survivant en ce noeud sera donc le chemin (a, c, b, a).
A l'instant $i = 4$, il reste 4 chemins survivants qui convergent respectivement :
- vers l'état a avec une distance de 2
- vers l'état b avec une distance de 3
- vers l'état c avec une distance de 1
- vers l'état d avec une distance de 3.

La séquence la plus vraisemblablement émise par le codeur est donc celle qui correspond au chemin survivant qui converge vers l'état c à $i = 4$ c'est-à-dire le chemin (a, c, b, a, c). Ce chemin correspond à l'émission par le codeur de la séquence 11101111 soit la séquence 1001 à l'entrée du codeur : le décodeur de Viterbi a corrigé l'erreur survenue dans la transmission.

Le décodage à entrées souples en utilisant l'algorithme de Viterbi améliore en théorie les performances d'environ 2 dB par rapport au décodage en décision dure. En pratique, il est nécessaire d'effectue une quantification des entrées sur 2^b niveaux. avec $b = 3$ ou $b = 4$.

Le calcul des performances théoriques des codes convolutifs avec décodage à décision dure ou souple s'obtient simplement à partir de la fonction de transfert du code [30].

10.5.1 Complexité du décodeur de Viterbi

Pour un codeur convolutif de rendement k/n et composé de M mémoire internes, on a 2^M états internes : l'utilisation de l'algorithme de Viterbi nécessite de conserver à chaque étape 2^M chemins survivants et métriques cumulées associées. A chaque étape 2^k chemins convergent vers chaque

noeud ; pour chacun de ces chemins, on doit calculer la métrique cumulée afin de déterminer le chemin survivant. En conséquence le nombre de calculs de métrique cumulée à effectuer à chaque étape est égal à 2^{k+M}. Ce calcul croit exponentiellement avec k et M ce qui limite l'utilisation de l'algorithme de Viterbi à de faibles valeurs de k et M.

En pratique, il n'est pas nécessaire d'attendre que l'ensemble des symboles émis par le codeur soit reçu pour commencer le décodage : en effet, on observe qu'à partir d'un certain temps (5 fois M environ), tous les chemins survivants tendent à converger vers le même chemin. Il est donc possible à l'instant i d'estimer le symbole émis à l'instant $i - 5M$.

10.5.2 Poinçonnage des codes convolutifs

Il est possible à partir d'un code convolutif de rendement k/n de réaliser des codes convolutifs de rendement supérieur en supprimant certains symboles en sortie du codeur [31] [5].

Cette technique permet de modifier le rendement du code convolutif sans ajouter de complexité au décodeur. Pour comprendre le principe du poinçonnage, nous allons considérer l'exemple d'un codeur de rendement 2/3 obtenu à partir du codeur de rendement 1/2 de l'exemple 1.

Le poinçonnage s'obtient par exemple en supprimant un symbole sur 4 en sortie du codeur de rendement 1/2.

A la réception on ajoute à la place de chaque symbole manquant un symbole nul.

Le diagramme en treillis de ce codeur perforé de rendement 2/3 est représenté sur la figure 10.12.

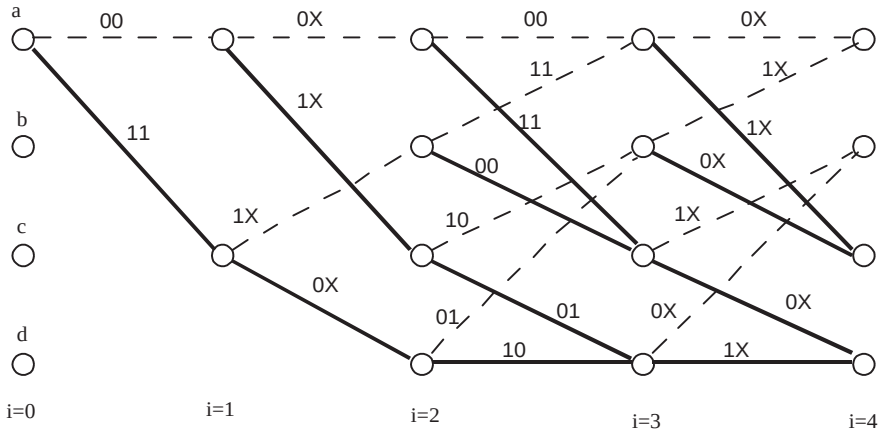


FIG. 10.12 – Diagramme en treillis d'un codeur convolutif poinçonné.

La distance minimale du code convolutif de rendement 1/2 et $M = 2$ est égale à 5 ; elle est réduite à 3 dans le cas du code poinçonné de rendement 2/3. Il faut préciser qu'il n'existe pas de code de rendement 2/3 avec $M = 2$ ayant une distance minimale supérieure à 3.

Chapitre 11

Applications des codes correcteurs d'erreurs

11.1 Communications spatiales

Dans toutes les communications spatiales la modulation utilisée est une modulation à 2 états de phase (BPSK).

• Mission Mariner en 1969 et Viking (Mars)
code en bloc Reed-Muller (32,6,16), décodeur à entrées pondérées

• Mission Pionner 9 en 1969 (soleil) Pionner 10 en 1972 (jupiter) Pionner 11 en 1973, Pionner 12 (vénus), Helios A et B (soleil)
code convolutif rendement $1/2$ $K_c = 32$, décodeur séquentiel à entrées pondérées

• Mission Voyager 1 et 2 en 1977 (jupiter et saturne) et standard CCSDS
code convolutif (163,171) rendement $1/2$, $K_c = 7$, décodeur de Viterbi à entrées pondérées

• Satellites Globalstar
code convolutif rendement $1/2$, $K_c = 9$

• Standard CCSDS et Mission Voyager
code convolutif (163,171) rendement $1/2$, $K_c = 7$, code Reed-Solomon (255,223,33)

• Mission Galileo en 1990 (jupiter)
code convolutif rendement $1/4$, $K_c = 15$, $d_{min} = 35$, the Big Viterbi Decoder (BVD)

• Mission Cassini (saturne)
code convolutif rendement $1/6$ $K_c = 15$, décodeur de Viterbi

11.2 Diffusion Audio et Télévision Numérique

• Télétext :
code de Hamming étendu (8,4)

• FM Digital Audio Broadcast (DAB) :
code cyclique difference set (273,191), décodage à logique majoritaire (entrées dures ou pondérées)

• Digital Audio Broadcast (DAB) pour le système audio DVB et MPEG :

code convolutif récursif rendement 1/2 $K_c = 5$, perforé (grand choix pour permettre une dégradation progressive des performances)

• système par satellite DirectTV, Digital Video Broadcast (DVB) satellite :
modulation QPSK + code convolutif (163,171) rendement 1/2 , $K_c = 7$, perforation 3/4,4/5, 5/6 et 7/8 , code Reed-Solomon (204,188,17)

• DVB terrestre :
modulation multiporteuses OFDM, QAM64 + code convolutif (163,171) rendement 1/2 , $K_c = 7$, perforation 3/4,4/5, 5/6 et 7/8 , code Reed-Solomon (204,188,17)

11.3 Transmission de données

• V29 en 1976 2400 b/s , half duplex :
modulation QAM16, 4b/symb

• V32 en 1984 9600 b/s full duplex :
modulation codée en treillis 32-CROSS 2 bits non codés et 2 bits codés avec code convolutif (3,2) $K_c = 4$ 4b/symb

• V33 en 1986 14400 b/s full duplex :
modulation codée en treillis 128-CROSS 4 bits non codés et 2 bits codés avec code convolutif (3,2) $K_c = 5$ 6b/symb

• V34 en 1994 33600 b/s full duplex :
modulation codée en treillis 4D, jusqu'à 1664 points, 8 bits non codés et 2 bits codés avec code convolutif (3,2) $K_c = 5$ 10b/symb

11.4 Stockage de données

• disque dur magnétique :
TBD

• CD audio :
RS(28,24) + RS(32,28) soit un rendement R=3/4

• CD ROM :
code produit (1170,1032) composé de codes lignes RS(26,24) et de codes colonnes RS(45,43)dans $GF(2^8)$

11.5 Radio communications

• GSM :
modulation GMSK, code convolutif rendement 1/2, $K_c = 5$

11.6 Transfert de fichiers

• TCP/IP :
CRC 16 bits sur l'entête, ARQ
• HDLC :
CRC 16 bits du CCITT ou CRC 32 bits optionnel, ARQ

Annexe 1

Dans cette annexe, nous allons démontrer quelques résultats utiles pour la démonstration géométrique de la capacité d'un canal à bruit blanc additif gaussien.

Soit $\mathbf{n} = (n_1, n_2, \ldots, n_D)$ le vecteur bruit composé de D composantes indépendantes. La densité de probabilité de chacune de ces composantes est gaussienne :

$$p(n_i) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{n_i^2}{2\sigma^2}\right) \tag{11.1}$$

La densité de probabilité du vecteur $\mathbf{n}$ s'exprime comme le produit des D densités de probabilité $p(n_i)$:

$$p(\mathbf{n}) = \frac{1}{(2\pi\sigma^2)^{D/2}} \exp\left(-\frac{\sum_{i=1}^{D} n_i^2}{2\sigma^2}\right) \tag{11.2}$$

Soit $r = \sqrt{\sum_{i=1}^{D} n_i^2}$ la norme de $\mathbf{n}$. Comme la variance de n_i est $E(n_i^2) = \sigma^2$, la moyenne de r^2 est donnée par

$$\begin{aligned} E(r^2) &= E(n_1^2 + n_2^2 + \cdots + n_D^2) \\ &= E(n_1^2) + E(n_2^2) + \cdots + E(n_D^2) \\ &= D.\sigma^2 \end{aligned} \tag{11.3}$$

Et la variance de n_i^2 est

$$\begin{aligned} var(n_i^2) &= E(n_i^4) - E(n_i^2)^2 \\ &= \int_{-\infty}^{+\infty} n_i^4 p(n_i^2) dn_i - E(n_i^2)^2 \\ &= \int_{-\infty}^{+\infty} \frac{n_i^4}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{n_i^2}{2\sigma^2}\right) dn_i - \sigma^4 \\ &= 3\sigma^4 - \sigma^4 \\ &= 2\sigma^4 \end{aligned} \tag{11.4}$$

Pour D grand, en utilisant le théorème de la limite centrale, on montre que la variance de r^2 est égale à $2\sigma^4 D$. Ainsi lorsque D tend vers l'infini, la norme r est concentrée autour de $\sqrt{D}.\sigma$

Annexe 3 : Calcul de la densité spectrale de puissance d'un signal émis en bande de base

Le signal à émettre dans le cas d'une transmission en bande de base s'écrit :

$$x(t) = \sum_{k=-\infty}^{\infty} a_k g(t - kT) \tag{11.5}$$

où $\{a_k\}$ est la séquence discrète à transmettre de moyenne $m_a = E(a_k)$ et dont la fonction d'autocorrélation $\phi_{aa}(i)$ est égale à $E(a_{k+i}a_k^*)$.

La fonction d'autocorrélation de $x(t)$ est donnée par :

$$R_{xx}(t+\tau, t) = E\left[x(t+\tau)x^*(t)\right] \tag{11.6}$$

La notation * signifie complexe conjugué (ce qui permet de traiter aussi les cas où $x(t)$ est un signal complexe)

Le signal $x(t)$ défini ci dessus est aléatoire mais n'est pas stationnaire. Il est cyclostationnaire d'ordre 2 car sa fonction d'autocorrélation est invariante à une translation de l'origine des temps proportionnelle à T.

Pour calculer sa densité spectrale de puissance, on suppose que l'origine des temps est une variable aléatoire suivant une loi uniforme entre 0 et T. Ainsi, au lieu de calculer $R_{xx}(t+\tau, t)$, nous calculerons la moyenne $\bar{R}_{xx}(\tau)$:

$$\bar{R}_{xx}(\tau) = \frac{1}{T}\int_{-T/2}^{+T/2} R_{xx}(t+\tau, t)dt \tag{11.7}$$

En développant $R_{xx}(t+\tau, t)$ on obtient :

$$\begin{aligned}
\bar{R}_{xx}(\tau) =& \frac{1}{T}\int_{-T/2}^{+T/2} \sum_{k=-\infty}^{+\infty}\sum_{i=-\infty}^{+\infty} E(a_i a_k^*) g^*(t-kT)g(t+\tau-iT)dt \\
=& \frac{1}{T}\int_{-T/2}^{+T/2} \sum_{k=-\infty}^{+\infty}\sum_{i=-\infty}^{+\infty} \phi_{aa}(i-k) g^*(t-kT)g(t+\tau-iT)dt \\
=& \sum_{i'=-\infty}^{+\infty} \phi_{aa}(i') \sum_{k=-\infty}^{+\infty} \frac{1}{T}\int_{-T/2}^{+T/2} g^*(t-kT)g(t+\tau-i'T-kT)dt \\
=& \sum_{i'=-\infty}^{+\infty} \phi_{aa}(i') \sum_{k=-\infty}^{+\infty} \frac{1}{T}\int_{-T/2+kT}^{+T/2+kT} g^*(t)g(t+\tau-i'T)dt
\end{aligned} \tag{11.8}$$

Dans l'avant dernière ligne de calcul,nous avons posé $i' = i - k$.

La fonction d'autocorrélation de $g(t)$ est :

$$R_{gg}(\tau) = \int_{-\infty}^{+\infty} g^*(t)g(t+\tau)dt \tag{11.9}$$

Finalement on obtient alors :

$$\bar{R}_{xx}(\tau) = \frac{1}{T}\sum_{i'=-\infty}^{+\infty} \phi_{aa}(i')R_{gg}(\tau - i'T) \tag{11.10}$$

$$\text{Soit} \qquad \gamma_{AA}(f) = \sum_{i=-\infty}^{+\infty} \phi_{aa}(i)e^{-j2\pi fiT} \tag{11.11}$$

$$\text{Et} \qquad |G(f)|^2 = TF(R_{gg}(\tau)) \tag{11.12}$$

On obtient finalement :

$$\boxed{\gamma_{XX}(f) = \frac{1}{T}|G(f)|^2\gamma_{AA}(f)} \tag{11.13}$$

Cette formule est appelée formule de Bennett.

Cas particulier 1 : considérons le cas où les symboles a_k sont indépendants, de moyenne nulle $E(a_k) = 0$ et de variance $E((a_k)^2) = \sigma^2$
la fonction d'autocorrélation $\phi_{aa}(i)$ est égale à

$$\phi_{aa}(i) = \begin{cases} \sigma^2 & \text{si} \quad i = 0 \\ 0 & \text{sinon} \end{cases} \tag{11.14}$$

Alors, $\gamma_{AA}(f) = \sigma^2$
On a donc :

$$\gamma_{XX}(f) = \frac{\sigma^2}{T}|G(f)|^2 \tag{11.15}$$

Cas particulier 2 : considérons le cas où les symboles a_k sont indépendants, de moyenne non nulle μ et de variance centrée $E((a_k)^2) = \sigma^2$
la fonction d'autocorrélation $\phi_{aa}(i)$ est égale à

$$\phi_{aa}(i) = \begin{cases} \mu^2 + \sigma^2 & \text{si} \quad i = 0 \\ \mu^2 & \text{sinon} \end{cases} \tag{11.16}$$

Alors,

$$\begin{aligned} \gamma_{AA}(f) &= \sum_{i=-\infty}^{+\infty} \phi_{aa}(i)e^{-j2\pi fiT} \\ &= \sigma^2 + \mu^2 \sum_{i=-\infty}^{+\infty} e^{-j2\pi fiT} \\ &= \sigma^2 + \frac{\mu^2}{T}\sum_{i=-\infty}^{+\infty} \delta\Big(f - \frac{i}{T}\Big) \end{aligned} \tag{11.17}$$

La dernière ligne se justifie car la sommation peut être vue comme la série de Fourier d'un train d'impulsion d'amplitude et de largeur $1/T$:

$$TF\Big(\sum_{i=-\infty}^{+\infty}\delta\Big(t-\frac{i}{T}\Big)\Big)=\frac{1}{T}\sum_{i=-\infty}^{+\infty}\delta\Big(f-\frac{i}{T}\Big) \tag{11.18}$$

Finalement la densité spectrale de puissance s'écrit alors :

$$\gamma_{XX}(f)=\frac{\sigma^2}{T}|G(f)|^2+\frac{\mu^2}{T^2}\sum_{i=-\infty}^{+\infty}\Big|G\Big(\frac{i}{T}\Big)\Big|^2\delta\Big(f-\frac{i}{T}\Big) \tag{11.19}$$

On peut observer que si la moyenne μ n'est pas nulle, la densité spectrale de puissance est composée de deux termes. Le premier terme correspond à un spectre continu et est directement lié à la réponse $G(f)$. Le second terme est un spectre de raie à toutes les fréquences multiples de $1/T$.

Il faut préciser que dans un système de communication pratique, il est souhaitable d'avoir une moyenne nulle.

Annexe 4 : Signaux aléatoires

Généralités

On appelle $m_x(t)$ la moyenne d'un signal aléatoire $x(t)$:

$$m_x(t) = E\big[x(t)\big] \tag{11.20}$$

On appelle $R_{xx}(\tau)$ sa fonction d'autocorrélation :

$$R_{xx}(t_1, t_2) = E\big[x(t_1)x(t_2)\big] \tag{11.21}$$

Le signal $x(t)$ est stationnaire au second ordre ou au sens large si :

1. la moyenne $m_x(t)$ est constante
2. la fonction d'autocorrélation satisfait $R_{xx}(t_1, t_2) = R_{xx}(t_1 + t, t_2 + t)\ \forall t$

On note alors simplement

$$R_{xx}(\tau) = E\big[x(t)x(t-\tau)\big] \tag{11.22}$$

Dans ce cas, la densité spectrale de puissance $\gamma_{xx}(f)$ s'obtient par :

$$\begin{aligned}\gamma_{xx}(f) &= TF(R_{xx}(\tau)) \\ &= \int_{-\infty}^{+\infty} R_{xx}(\tau)e^{-j2\pi f\tau}d\tau\end{aligned} \tag{11.23}$$

Réciproquement, la fonction d'autocorrélation ($R_{xx}(\tau)$) se détermine à partir de la densité spectrale de puissance comme suit :

$$\begin{aligned}R_{xx}(\tau) &= TF^{-1}(\gamma_{xx}(f)) \\ &= \int_{-\infty}^{+\infty} \gamma_{xx}(f)e^{+j2\pi f\tau}df\end{aligned} \tag{11.24}$$

De façon générale, la moyenne et la fonction d'autocorrélation d'un signal stationnaire sont estimées à partir d'un ensemble de réalisations du signal $x(t)$. Lorsque la moyenne temporelle tend vers cette moyenne, on dit que le signal aléatoire est ergodique. Une seule réalisation du processus aléatoire $x(t)$ suffit pour estimer la moyenne et la fonction d'autocorrélation. Les signaux aléatoires que nous sommes amenés à rencontrer dans les communications numériques sont en général stationnaire et ergodique au second d'ordre.

Si le signal est discret (par exemple suite à échantillonnage d'un signal continu aléatoire $x(t)$ à la fréquence $\frac{1}{T}$, $= x_n = x(nT)$, la fonction d'autocorrélation $R_{xx}(\tau)$ n'est définie qu'aux instants discrets $\tau = nT$ et la densité spectrale de puissance devient :

$$\begin{aligned}\gamma_{xx}(f) &= TFD(R_{xx}(\tau)) \\ &= \sum_{-\infty}^{+\infty} R_{xx}(\tau)e^{-j2\pi f\tau}\end{aligned} \tag{11.25}$$

Puissance

La puissance de $x(t)$ est définie par :

$$\begin{aligned}P &= \int_{-\infty}^{+\infty} \gamma_{xx}(f)df \\ &= R_{xx}(0) \\ &= E\big[x(t)^2\big]\end{aligned} \tag{11.26}$$

Si le signal est discret, on a :

$$P = E\big[x_n^2\big] \tag{11.27}$$

Energie

L'énergie d'un signal aléatoire $x(t)$ est :

$$\mathcal{E} = \int_{-\infty}^{+\infty} x(t)^2 dt \tag{11.28}$$

Si le signal est discret, on a :

$$\mathcal{E} = \sum_{n=-\infty}^{+\infty} x_n^2 \tag{11.29}$$

Cas du bruit blanc

Soit un bruit blanc centré $b(t)$ défini par sa densité spectrale de puissance bilatérale $\gamma_{bb}(f) = \frac{N_0}{2}$. Sa fonction d'autocorrélation $R_{bb}(\tau)$ est égale à

$$\begin{aligned}R_{bb}(\tau) &= \int_{-\infty}^{+\infty} \gamma_{bb}(f)\exp(2j\pi f\tau)df \\ &= \frac{N_0}{2}\delta(\tau)\end{aligned} \tag{11.30}$$

où $\delta(.)$ représente l'impulsion de Dirac.

On notera que la densité spectrale de puissance s'exprime en V^2/Hz et que $\delta(\tau)$ s'exprime en s^{-1}.

Le calcul de la puissance du bruit blanc défini ci-dessus est infinie :

$$\begin{aligned}P &= \int_{-\infty}^{+\infty} \gamma_{bb}(f)df \\ &= \int_{-\infty}^{+\infty} \frac{N_0}{2}df \\ &= +\infty\end{aligned} \tag{11.31}$$

Un tel signal n'existe pas en pratique.

Considérons maintenant le cas du bruit blanc dans une bande de fréquence limitée B. On a :

$$\gamma_{bb}(f) = \begin{cases} \frac{N_0}{2} & \text{si} \quad -B \leq f \leq +B \\ 0 & \text{sinon} \end{cases} \tag{11.32}$$

La fonction d'autocorrélation de ce signal est :

$$\begin{aligned} R_{bb}(\tau) &= \int_{-B}^{+B} \frac{N_0}{2}(f) e^{+j2\pi f\tau} df \\ &= \frac{N_0}{2} \frac{sin(2\pi B\tau)}{\pi\tau} \end{aligned} \tag{11.33}$$

Dans ce cas, la puissance N dans la bande B est égale à :

$$N = \int_{-B}^{+B} \frac{N_0}{2} df = N_0 B \tag{11.34}$$

Si ce bruit blanc dans la bande B est gaussien, sa densité de probabilité sera la suivante :

$$p(b) = \frac{1}{\sqrt{2\pi N}} \exp\left(-\frac{b^2}{2N}\right) \tag{11.35}$$

Bibliographie

[1] G. Battail. *"Théorie de l'information"*. Edition Masson, 1997.

[2] E. R. Berlekamp. *"Algebraic coding theory"*. Edition Mc Graw-Hill,New York, 1968.

[3] C. Berrou, A. Glavieux, P. Thitimajshima. *"Near Shannon limit error correcting coding and decoding : Turbo-codes"*. Proc. of the 1993 Int. Conf. on Comm., Geneva, Switzeland, pp. 1064–1070, Mai. 1993.

[4] R.C Bose, D. K. Ray-Chaudhuri *"On a class of error correcting binary group codes"*. Inform. Control, vol.3, pp 68–79, mars 1960.

[5] J.B. Cain, G. C. Clark, J. M. Geist *"Punctured convolutional codes of rate $n-1/n$ and simplified maximum likelihood decoding"*. IEEE Trans. Inform. Theory **25**, pp. 97–100, Janvier 1979.

[6] D. Chase. *"A class of algorithms for decoding block codes with channel measurement information"*. IEEE Trans. on Inform. Theory. **18**, pp. 170–182, Jan. 1972.

[7] G. Cohen *"Codes correcteurs d'erreurs"*. Edition Masson, 1990.

[8] T. M. Cover et J. A. Thomas. *"Elements of information theory"*. Edition Wiley, 1991.

[9] P. Elias. *"Error-free coding"*. IRE Trans. Inform. Theory, pp. 29–37, 1954.

[10] G. D. Forney. *"Concatenated codes"*. MIT Press, Cambridge,Mass., 1966.

[11] G. D. Forney. *"Convolutional codes I : Algebraic structure"*. IEEE Trans. Inform. Theory **16**(6), pp. 720–738, 1970.

[12] G. D. Forney. *"The Viterbi algorithm"*. Proceedings of the IEEE **61**(3), pp. 268–278, Mars 1973.

[13] R. G. Gallager. *"Low density parity-check codes"*. MIT Press, Cambridge,Mass., 1963.Trans. Inform. Theory **25**, pp. 97–100, Janvier 1979.

[14] R. G. Gallager. *"Information theory and reliable communication"*. Edition Wiley, 1968.

[15] A. Hocquenghem. *"Codes correcteurs d'erreurs"*. Chiffres, vol.2, pp 147–156, sept. 1959.

[16] D. A. Huffman. *"A method for the construction of minimum redundancy codes"*. Proc. IRE, vol. 40, pp. 1098–1101, septembre 1952.

[17] R. Johannesson, K. S. Zigangirov. *"Fundamentals of convolutionnal coding"*. IEEE Press, Piscataway, NJ, USA, 1999.

[18] F. R. Kschischang, B. J. Frey, H. A. Loeliger. *"Factor graphs and the sum-product algorithm"*. IEEE Trans. Inform. Theory **47**(2), pp. 498–519, Feb. 2001.

[19] R. J. McEliece, E. R. Rodemich, H. C. Rumsey, L. R. Welch. *"New upper bounds on the rate of a code via the Delsarte-MacWilliam"*. Proc. IEEE Trans. on Info. theory, vol. IT-23, pp.157–166, 1977.

[20] D. J. MacKay. *"Information Theory, Inference and Learning Algorithms"*. Téléchargeable sur le site htt ://www.inference.phy.cam.ac.uk/mackay

[21] F. J. MacWilliams, N. J. A. Sloane *The theory of error correcting codes* North Holland Elsevier, Amsterdam, The Netherlands, 1977

[22] J. L. Massey *"Shift register synthesis and BCH decoding"*. IEEE Trans. Inform. Theory **6**, pp. 459–470, Janvier 1960.

[23] W. W. Peterson, E. J. Weldon *Error correcting codes* MIT Press, Cambridge, MA, 1972

[24] J. G. Proakis *"Digital communications"*. McGraw-Hill,Boson,USA, 4^{eme} édition, 2001.

[25] I. S. Reed, G. Solomon. *"Polynomial codes over certain finite fields"*. Journal SIAM, vol.8, pp 300–304, 1960.

[26] J. J. Rissanen. *"Generalized Kraft inequality and arithmetic coding"*. IBM Journal Research and Dev., vol. 20, No. 3,pp. 198–203, mai 1976.

[27] C. Shannon. *"A mathematical theory of communication"*. Bell Syst. Tech. J., vol. 27, pp. 623–659 et pp. 623–656, juillet et octobre 1948. Téléchargeable sur le site http ://www.math.psu.edu/gunesh/Entropy/shannon.ps ou http ://cm.bell-labs.com/cm/ms/what/shannonday/paper.html

[28] B. Sklar. *"A primer on Turbo Code concepts"*. IEEE Communications Magazine , pp. 94–102, Dec. 1997.

[29] A. J. Viterbi. *"Error bounds for convolutional codes and an asymptotically optimum decoding algorithm"*. IEEE Trans. Inform. Theory **13**, pp. 260–269, Avril 1967.

[30] A. J. Viterbi, J. K. Omura. *"Principes des communications numériques"*. Edition Dunod, CNET-ENST, 1982.

[31] Y. Yasuda, K. Kashiki, H. Hirata. *"High rate punctured convolutionnal codes for soft-decision Viterbi decoding"*. IEEE trans. on Communications **32**, pp. 315–319, Mars 1984.

[32] J. Ziv et A. Lempel. *"Compression of individual sequences via variable rate coding"*. Proc. IEEE Trans. on Info. theory, vol. IT-24, No. 5,pp. 530–536, septembre 1978.

Table des matières