# Introduction à la théorie des codes

André Leroy

Ces notes doivent servir aux étudiants comme support et ne remplacent pas le cours.
Section 1 Introduction.
Section 2 Théorie de l'information.
Section 3 Distance de Hamming.
Section 4 Détection et correction d'erreurs.
Section 5 Un peu de probabilité.
Section 6 Quelques codes courants.
Section 7 codes linéaires.
Section 8 Décodage des codes linéaires.
Section 9 Codes de Hamming
Section 10 Codes cycliques.
Section 11 Gap et Guava.

## 1 Introduction

La théorie des codes correcteurs d'erreurs a pour but la création de codes capables de détecter et éventuellement de corriger des erreurs survenus lors de la transmission d'un message. Elle a pour base théorique la théorie de l'information qui a véritablement commencé par l'article de Claude Shannon (1948) dont on parlera très brièvement ci-dessous.

## 2 Théorie de l'information

L'envoi d'une information se fait en utilisant un alphabet $A$ qui est simplement un ensemble fini de symboles. Une chaîne finie de symboles de $A$ est appelé un mot. L'information à transmettre est encodée sous forme de mots appelés mots codes. Ces mots ont tous la même longueur (codage par blocs de même longueur). La transmission a lieu sur un canal bruité et des erreurs peuvent apparaître. Pour pouvoir détecter et éventuellement corrigé les erreurs on introduit de la redondance. Les mots du messages sont de longueur $k$ et les mots encodés et transmis sont de longueur $n \geq k$. Le rendement d'un code est le rapport entre la longueur du message $k$ et la longueur $n$ du message codé et transmis : $R = k/n$. Il semble naturel de penser qu'un code qui corrige les erreurs doive être de rendement nul... Shannon (1948) Non : Il existe une valeur limite du rendement, notée C, que l'on appelle la capacité du canal. Si l'on veut émettre avec un rendement $R$ supérieur à $C$, on est sûr de faire des erreurs : le bruit ne pourra être corrigé. En revanche, pour $R < C$, il

est possible de construire un codage de rendement $R$ et corrigeant toutes les erreurs. Pour un canal binaire symétrique la capacité et donnée par la formule

$$C = 1 + plog_2 p + (1-p)log_2(1-p).$$

par exemple pour $p = 0.01$ la capacité est d'environ $0,91921$. Le théorème de Shannon est en réalité un résultat asymptotique, nous n'en donnerons pas l'énoncé précis ici.

# 3 Codes et distance de Hamming

**Définition 3.1.** On appellera **mot** (de longueur $n$) une suite $b_1 b_2 \dots b_n$ où pour tout $i \in \{1, \dots, n\}$, $b_i \in A$ ($A$ est l'alphabet. Un code $C$ est donc un ensemble de mots construits sur un alphabet $A$. Un codage par blocs est un code tel que tous les mots ont la même longueur c'est donc un sous ensemble de $A^n$ où $A$ est l'alphabet du code. Dans ce cours on ne considérera que les codes bloc.

On notera $q = |A|$ et $M = |C|$.

**Proposition 3.2.** *Soit $C \subseteq A^n$. L'application $d : C \times C \longrightarrow \mathbb{N}$ définie par $d(a,b) = |\{i \,|\, a_i \neq b_i\}|$ où $a = (a_1, \dots, a_n) \in C$ et $b = (b_1, \dots, b_n) \in C$ est une distance sur $C$ c'est-à-dire qu'elle vérifie les propriétés suivantes :*

*(d1)* $d(a,b) \geq 0$, *légalité ayant lieu si et seulement si* $a = b$.

*(d2)* $d(a,b) = d(b,a)$.

*(d3)* $d(a,b) + d(b,c) \geq d(a,c)$.

**Définitions 3.3.**

(a) La distance définie dans la proposition 3.2 ci-dessus s'appelle la *distance de Hamming* du code $C$.

(b) La distance minimale du code est donnée par :

$$d := min|\{d(a,b) \,|\, a, b \in C\}|$$

Pour $n, d$ fixés on notera le nombre maximum de mots codes de longueur $n$ et de distance $d$ par $A_q(n,d)$. Des liens existent entre $n, d, M, q$ et $A_q(nd)$ ($q$ étant le cardinal de l'alphabet ; donc $q = 2$ pour un alphabet binaire). En voici quelques uns :

**Théorème 3.4.** *(a) Pour $n \geq 1$, $A_q(n,1) = q^n$ et $A_q(n,n) = q$.*

*(b) Pour $n \geq 2$, $A_q(n,d) \leq qA_q(n-1,d)$.*

*(c) Pour un code binaire $A_2(n, 2t+1) = A_2(n+1, 2t+2)$.*

*(d) La borne de Gilbert-Varshamov : $A_q(n,d) \geq \frac{q^n}{\sum_{k=0}^{d-1} \binom{n}{k}(q-1)^k}$.*

*(e) La borne de Singleton* $A_q(n,d) \leq q^{n-d+1}$.

*(f) La borne d'empilement des sphères : pour* $t = \lfloor \frac{d-1}{2} \rfloor$ *on a* $A_q(n,d) \leq \frac{q^n}{\sum_{k=0}^{t} \binom{n}{k}(q-1)^k}$.

*(g) La borne de Plotkin : On pose* $\theta = \frac{q-1}{q}$, *si* $d \geq \theta n$, *alors* $A_q(n,d) \leq \frac{d}{d-\theta n}$.

Un code est dit *parfait* si l'égalité a lieu dans la formule d'empilement des sphères.

# 4 Détection et correction d'erreurs

Dans toute la suite du cours les codes seront des codes par blocs sur un alphabet binaire noté $\mathbb{B} = \{0,1\}$ ($\mathbb{B}$ est le corps à deux éléments souvent noté $\mathbb{F}_2$). Les mots codés seront de longueur $n$ ce seront des éléments de l'espace $\mathbb{B}^n$.

Calculons la distance de Hamming entre $x = 010011$ et $y = 011101$. cette distance vaut 3. On peut la calculer en faisant la somme binaire des deux mots $x$ et $y$ et en comptant le nombre de 1. Cette somme binaire est parfois notée $\oplus$. Détaillons : Pour la longueur $n = 1$ on a donc

$$0 \oplus 0 = 0 \quad 0 \oplus 1 = 1 \quad 1 \oplus 0 = 1 \quad 1 \oplus 1 = 0$$

L'addition dans $\mathbb{B}^n$ se fait composante par composante. Par exemple si $x = 10001101$ et $y = 00110100$ $x \oplus y = 10111001$. On remarque que $d(x,y)$ est le nombre de "1" dans $x \oplus y$, on a donc par exemple $d(x,y) = 5$.

**Définitions 4.1.**

Soit $x \in \mathbb{B}^n$. On appelle poids de $x$ et on note $w(x)$, le nombre de composantes égales à 1 dans $x$.

La distance minimale d'un code $C \subseteq \mathbb{B}^n$, est définie par $d(C) := Min\{d(x,y) \,|\, x,y \in C\}$.

Les propriétés suivantes sont faciles à établir

**Proposition 4.2.**

$d(x,y) = w(x \oplus y)$.

$d(x \oplus z, y \oplus z) = d(x,y)$.

Supposons maintenant que l'on encode un message et qu'on le transmette sur un canal bruité. Afin de corriger les erreurs de transmission éventuelles on introduit de la "redondance" (i.e. des informations supplémentaires que doivent satisfaire tous les mots codés). Si le mot initial est de longueur $k$ le mot encodé et transmis est de longueur $n \geq k$. On appelle $(k,n)$ codage toute injection $e$ de $\mathbb{B}^k$ dans $\mathbb{B}^n$. On appelle ensemble des mots codes l'ensemble image $e(\mathbb{B}^k)$ de cette application. Si $m \in \mathbb{B}^k$ alors $e(m)$ est le mot code représentant $m$. On a donc le schéma suivant :

$$m \in \mathbb{B}^k \Longrightarrow m_t \in \mathbb{B}^n \Longrightarrow m_r \in \mathbb{B}^n$$

Où $m$ est le mot initial du message, $m_t$ est le mot à transmettre = mot transmis et $m_r$ est le mot reçu. La première flèche correspond à l'encodage (donc l'encodage est une application $e : \mathbb{B}^k \longrightarrow \mathbb{B}^n$). La deuxième flèche correspond à la transmission au travers du canal.

**Définition 4.3.** On appelle taux de codage le rapport $k/n$. On appelle vecteur d'erreur le vecteur $m_t \oplus m_r$

**Exemple 4.4.** Un code très utilisé est le contrôle de parité. C'est un $(k, k+1)$-code basé sur l'application $e : \mathbb{B}^k \longrightarrow \mathbb{B}^{k+1} b_1 \dots b_k \mapsto b_1 \dots b_k b_{k+1}$, où $b_{k+1}$ vaut 0 si $w(b_1 \dots b_k)$ est pair et $b_{k+1} = 1$ si $w(b_1 \dots b_k)$ est impair. Ce codage ne permet de détecter que les erreurs qui se sont produites en nombre impair. Mais même si on détecte qu'il y a eu une erreur on ne peut la corriger.Le taux de codage est "bon" (proche de 1) mais la détection d'erreur est assez limitée.

Si on utilise la méthode du maximum de vraisemblance (on décode un mot reçu comme étant le mot (l'un des ) mot code le plus proche) on essaie d'avoir une grande distance entre les mots code de façon à pouvoir détecter et éventuellement corriger un maximum de nombre de bits mal transmis.

**Théorème 4.5.** *(a) Si $d(C) \geq s+1$ alors $C$ peut détecter au plus $s$ erreurs.*
*(b) Si $d(C) \geq 2t+1$ alors $C$ peut corriger au plus $t$ erreurs.*

*Démonstration.* (a) Si $d(C) \geq s+1$. Soit $x$ le mot code envoyé et $y$ le mot reçu. Si il y a eu au plus $s$ erreurs alors $d(x, y) \leq s$. Donc $y$ ne peut pas être un mot code (puisque $d(x, y) < d(C)$) et l'erreur est détectée.
(b) Supposons $d(C) \geq 2t+1$. Soit $x$ le mot code envoyé et y le mot reçu. Supposons qu'il y ait eu au plus $t$ erreurs. On a alors $d(x, y) \leq t$. Si $x'$ est un autre mot code différent de $x$ alors $d(x', y) + t \geq d(x', y) + d(y, x) \geq d(x', x) \geq d(C) \geq 2t+1$. On conclut que $d(x', y) \geq t+1$. Ceci signifie que $x$ est l'unique mot code $c$ tel que $d(c, x) \leq t$. □

**Définitions 4.6.** Soit $e : \mathbb{B}^k \to \mathbb{B}^n : m \mapsto m_r$ un $(n, k)$-code et soit $p \in \{1, \dots, n\}$.

(a) On dit que $e$ est un $p$-détecteur si :

$$\forall m \in \mathbb{B}^k, (1 \leq d(e(m), m_r) \leq p) \Rightarrow m_r \notin e(\mathbb{B}^k)$$

(b) On appelle décodage associé à $e$, n'importe quelle fonction $f : \mathbb{B}^n \to \mathbb{B}^k$ telle que

$$f \cdot e = 1_{\mathbb{B}^k}$$

(c) Soit $1 \leq p \leq n$. On dit qu'un décodage $f : \mathbb{B}^n \to \mathbb{B}^k$ est $p$-correcteur si chaque fois que le nombre d'erreurs de transmission ne dépasse pas $p$ alors le mot reçu est décodé correctement :

$$\forall m \in \mathbb{B}^k : (0 \leq d(e(m), m_r) \leq p) \Rightarrow f(m_r) = m.$$

Le code $C$ résultant d'un codage $e : \mathbb{B}^k \Rightarrow \mathbb{B}^n$ est $C = e(\mathbb{B}^k) \subset \mathbb{B}^n$. On a donc comme conséquence de (a) dans le théorème 4.5 le corollaire suivant.

**Corollaire 4.7.** *(a) Un codage $e : \mathbb{B}^k \to \mathbb{B}^n$ est $t$-détecteur si et seulement si $t < d(e) = d(e(\mathbb{B}^k))$.*

*(b) Soit $0 \leq t \leq n$ et $f$ un décodage associé par maximum de vraisemblance à un codage $e : \mathbb{B}^k \to \mathbb{B}^n$. Le décodage est $t$-correcteur si et seulement si $t \leq \frac{d(e)-1}{2}$.*

**Exemples 4.8.** a) Soit le code $C \subset \mathbb{B}^5$ défini par : $C = \{11100, 10110, 00100, 01010\}$. Calculer la distance minimale du code, le nombre d'erreurs qu'il peut détecter, le nombre d'erreurs qu'il peut corriger
b) Idem pour le code défini par l'encodage $e$ :

| m | e(m) |
|---|---|
| 00 | 010101 |
| 01 | 101010 |
| 10 | 111000 |
| 11 | 001100 |

Quel est le taux de codage ?

*Le problème principal de la théorie des codes* : Un code $C$ est un $(n, M, d)$ code si la longueur des mots codes est $n$, $|C| = M$ et la distance minimale du code est $d$. Un bon code doit satisfaire les conditions suivantes :

1. $n$ doit être petit (rapidité et coût de transmission)
2. $M$ doit être grand (pour permettre une grande variété de messages)
3. $d$ doit être grand (pour détecter et éventuellement corrigés beaucoup d'erreurs)

Bien sur ces conditions sont conflictuelles. On a vu plus haut qu'il existait des bornes limitant, par exemple $M$ pour $n$ et $d$ donnés. Par exemple la borne de Hamming (borne d'empilement des sphères) donne dans le cas binaire ($q = 2$) pour $t = \lfloor \frac{d-1}{2} \rfloor$ l'inégalité $M \leq \frac{2^n}{\sum_{k=0}^{t} \binom{n}{k}}$.

Donc le problème principal de la théorie des codes est : *pour une longueur $n$ donnée et une distance $d$ fixée, trouver un code $C$ ayant un nombre de mots code maximal.*

# 5 Un peu de probabilité, canal binaire symétrique

Le canal est bruité le message transmis est altéré. On suppose que ce canal est symétrique c'est-à-dire que la probabilité d'erreur sur un bit est indépendante du contenu (0 ou 1). En outre le contenu d'un bit est indépendant du contenu des autres bits du mot code (canal sans mémoire). En faisant des statistiques sur un canal donné, on peut évaluer la probabilité de transmettre correctement un bit à travers ce canal. On notera $p$ la probabilité d'erreur sur un bit.

On suppose donc que :
1) Les erreurs qui apparaissent sur les différents bits sont indépendantes les unes des autres.
2) Chaque symbole de l'alphabet a la même probabilité $p < 1/2$ d'être en erreur.
3) Si un symbole $a \in A$ est mal transmis alors les $q - 1$ autres symboles ont la même probabilité d'avoir été envoyés.

Dans le cas d'un canal binaire (i.e. $|A| = 2$) on a donc :
La probabilité de transmettre correctement un mot de longueur $n$ donc $(1-p)^n$.
La probabilité que le mot contienne au moins une erreur est $1-(1-p)^n$.
La probabilité d'avoir exactement $i$ erreurs sur des bits bien spécifiés est $p^i(1-p)^{n-i}$.
La probabilité d'avoir $i$ erreurs est donc $\binom{n}{i}p^i(1-p)^{n-i}$.

**Exemples 5.1.**

1) Considérons le code obtenu par ajout d'un bit de parité : $e : \mathbb{B}^3 \longrightarrow \mathbb{B}^4$. On envoie donc sur le canal des mots de longueur 4. Lors du passage dans le canal il peut se produire des erreurs. Certaines seront détectées... Calculons la probabilité qu'un mot soit mal transmis sans que l'erreur ne soit détectée. On note $p$ la probabilité d'erreur sur un bit. On a successivement :
Probabilité qu'un mot soit mal transmis : $1-(1-p)^4$
La probabilité que le nombre d'erreurs de transmission soit 2 est $\binom{2}{4}p^2(1-p)^2$.
La probabilité que le nombre d'erreurs de transmission soit 4 est $p^4$.
La proportion de message comportant des erreurs de transmission qui ne sont pas détectées est donc
$$\frac{\binom{2}{4}p^2(1-p)^2 + p^4}{1-(1-p)^4}.$$
Si $p = 0,05$ il y a $7,3\%$ de messages qui comportent des erreurs non détectées.

2) On considère le code binaire qui consiste à tripler chaque bit à transmettre. C'est à dire que 0 est codé 000 et 1 est codé 111.
Si le mot reçu n'est pas un mot-code, le receveur est sur qu'une ou deux erreurs ont été commise(s).
Si le mot reçu est un mot code alors le mot a été correctement transmis ou il y a eu trois erreurs.
Ce codage sécurise davantage les informations mais envoie trois fois plus de bits que le code de parité.
La proportion de message comportant des erreurs non détectés est $\frac{p^3}{1-(1-p)^3}$.
Pour $p = 0,05$ la proportion de messages comportant des erreurs non détectées est d'environ $0,09\%$.

On peut redonner la définition d'un code parfait dans $\mathbb{B}^n$ : un code parfait est un code $t$-correcteur tel que l'ensemble code est une partition de $\mathbb{B}^n$ en boules de rayon $t$.

# 6 Quelques codes courants

Parfois le récepteur peut demander à l'émetteur de retransmettre l'information à moindre coût(ce n'est pas toujours possible). Il faut pour cela coder et décoder rapidement et détecter efficacement les erreurs. Pour décoder il suffit de commencer le mot codé par le mùot d'origine (codage systématique) puis d'ajouter des bits pour la détection d'erreurs.
**Codes EAN**

Le code EAN-13 (European Article Numbering) est un code à barres utilisé dans l'industrie et le commerce pour identifier de manière univoque les produits. Il est composé d'un numéro de 13 chiffres $c_{12} - -c_{11}c_{10}c_9c_8c_7c_6 - -c_5c_4c_3c_2c_1c_0$ et d'un motif graphique avec des barres noires et blanches pouvant être lues par un scanner optique. les chiffres $c_{12}$ à $c_1$ identifient l'objet.

Le dernier chiffre $c_0$ est calculé comme suit :

$$a = \sum_{i=1}^{6} c_{2i} \quad b = \sum_{i=1}^{6} c_{2i-1}$$

et

$$c_0 = 10 - (a + 3b) \mod 10.$$

Ce code permet de détecter les erreurs mais ne les corrige pas.

**Codes ISBN**

Le code ISBN (International Standard book number) est un code servant à identifier les livres dans le monde.

Il s'agit d'un numéro comportant 10 chiffres $c_{10}c_9c_8c_7c_6c_5c_4c_3c_2c_1$, structurés en quatre segments $A - B - C - D$ séparés par un tiret. Les neuf premiers $A - B - C$ identifient le livre :
$A$ : communauté linguistique.
$B$ : numéro de l'éditeur.
$C$ numéro d'ouvrage chez l'éditeur.

Le dernier chiffre est une clé de contrôle. ce chiffre est un nombre entier entre 0 et 9 ou la lettre $X$ représentant la nombre 10 Si $c$ représente la valeur de $D$ alors on doit avoir :

$$11 - \sum_{i=1}^{10} i \times c_i \equiv 0 \mod 11.$$

Exemple compléter la séquence $210 - 050 - 692$ pour obtenir un numéro ISBN correct. Le codes $EAN$-13 d'un livre commence toujours par 978 est suivi des 9 premiers choffres du code ISBN, le dernier chiffre étant la clé de contrôle calculée de la manière mentionnée ci dessus pour un code $EAN$-13. Dans le cas de l'exemple ci dessus, quel est son code barre $EAN - 13$ correspondant ?

**Carte bancaire**

Le numéro de carte bancaire appelé $LUHN10$, est un numéro à 16 chiffres

$$c_{16}c_{15}c_{14}c_{13}c_{12} - c_{11}c_{10}c_9c_8 - c_7c_6c_5c_4c_3c_2c_1$$

Pour détecter une erreur on multiplie les chiffres d'indices impairs par deux et on enlève 9 à ceux qui dépasse 9. Tous ces chiffres sont sommés aux autres chiffres d'indice pairs . La somme doit être multiple de 10. On peut également vérifier la validité de la carte au moyen

des 7 chiffres qui figurent au dos de la carte. Les quatre premiers sont les quatre derniers du numéro de la carte. Les trois derniers sont le résultat d'un calcul qui fait intervenir le numéro de la carte et la date d'expiration.

# 7 Codes Linéaires

### A) Généralités

Comme on l'a vu (notamment en TD) il est assez difficile de travailler avec des sous ensembles quelconques de $\mathbb{B}^n$...L'énumération systématique est souvent matériellement impossible et le manque de structure empêche de pouvoir faire des raisonnements généraux. De très nombreux codes intéressants sont obtenus en demandant que le code soit un sous espace vectoriel de $\mathbb{B}^n$. Faisons donc quelques rappels :

Pour simplifier on ne parlera que de codes binaires. Rappelons d'abord que le corps a deux éléments $\{0, 1\}$ est muni d'une addtion que l'on a noté $\oplus$ :

$$0 \oplus 0 = 0 \quad 0 \oplus 1 = 1 \quad 1 \oplus 0 = 1 \quad 1 \oplus 1 = 0.$$

Rappelons ensuite qu'un espace vectoriel de dimension $n$ sur le corps à deux éléments $\mathbb{F}_2$ (que nous avons noté $\mathbb{B}$) est simplement l'ensemble des mots de longueur $n$ sur l'alphabet $\{0, 1\}$. Cet ensemble est muni d'une addition définie par :

$$(u_1, u_2, \ldots, u_n) + (v_1, v_2, \ldots, v_n) = (u_1 \oplus v_1, u_2 \oplus v_2, \ldots, u_n \oplus v_n)$$

Si on pose $e_1 := (1, 0 \ldots, 0)$, $e_2 := (0, 1, \ldots, 0)$ et, pour $1 \leq i \leq n$, $e_i := (0 \ldots, 1, 0 \ldots, 0)$ où l'élément 1 apparaît en $i^{eme}$ position, alors tout élément de $v = (v_1, \ldots, v_n \in \mathbb{B}^n$ s'écrit de manière unique sous la forme $v = \sum_{i=1}^{n} \alpha_i e_i$. on dit que les éléments $e_1, \ldots, e_n$ forment une base de $\mathbb{B}^n$.

Un sous espace vectoriel $W$ de $\mathbb{B}^n$ est alors un sous ensemble non vide de $\mathbb{B}^n$ tel que si $u, v \in W$ alors $u + v \in W$.

**Définition 7.1.** Un code binaire de longueur $n$ est un sous espace vectoriel de $\mathbb{B}^n$.

**Remarque 7.2.** *La multiplication par un scalaire ne joue ici aucun rôle à cause du fait que l'on ne travaille qu'avec des codes binaires.*

**Exemples 7.3.** 1) Le code binaire de longueur 3 défini par $C = \{000, 011, 101, 110\}$ est linéaire.
2) Décrire le plus petit code linéaire de longueur 5 contenant les mots $11000, 01010, 10011$.
3) Idem pour les mots $11001, 11100, 101011, 01110$.

L'un des avantages des codes linéaires est de permettre un calcul facile de la distance d'un tel code.

Pour un élément $x = (x_1, \ldots, x_n) \in \mathbb{B}^n$ rappelons que $w(x)$ est le nombre d'entrées $x_i$ tels que $x_i \neq 0$. On appelle poids d'un code $C$, noté $w(C)$, le nombre

$$w(C) = min\{w(x) \mid x \in C, x \neq 0\}.$$

**Théorème 7.4.** *Si $C$ est un code linéaire alors $w(C) = d(C)$.*

*Démonstration.* Soit $x, y \in C$ tels que $d(x, y) = d(C)$. Alors $d(C) = d(x, y) = w(x - y) \geq w(C)$. Réciproquement si $x \in C$ est tel que $w(C)w(x) = w(x - 0) = d(x, 0) \geq d(C)$. □

**Remarque 7.5.** *En général si $C$ est un code et $|C| = M$, alors, pour calculer $d(C)$ on doit évaluer*

$$\binom{M}{2} = \frac{M(M-1)}{2}$$

*distances. Si le code est linéaire on doit simplement en calculer $M - 1$.*

### B) Matrices génératrices et matrices de contrôle

Tout espace vectoriel possède une base. Un code linéaire $C$ est donc caractérisé par la donnée d'une base. En effet si $b_1, \ldots, b_k \in \mathbb{B}^n$ forment une base de $C$ alors tout élément de $C$ est une somme :

$$\sum_{i=1}^{k} \alpha_i b_i, \quad \alpha_i \in \mathbb{B} = \{0, 1\}.$$

Donc le code $C$ est complètement déterminé par la matrice notée $G$ de type $k \times n$ formée par les vecteurs de la base $b_1, \ldots, b_k$. Posons, pour tout $1 \leq i \leq k$, $b_i = b_{i1} \ldots b_{ik}$. La matrice : $G$, appelée matrice génératrice du code est alors donnée par :

$$G = \begin{pmatrix} b_{11} & b_{12} & \ldots & b_{1k} \\ b_{21} & b_{22} & \ldots & b_{2k} \\ \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots \\ b_{k1} & b_{k2} & \ldots & b_{kn} \end{pmatrix}$$

Un mot code est alors simplement une combinaison linéaire des lignes de cette matrice.

**Exemple 7.6.** Donner tous les mots codes du code linéaire de longueur 7 engendré par les mots 1100110, 1110001, 1010100.

Du point de vue de l'encodage, un code linéaire provient simplement d'une fonction d'encodage $e : \mathbb{B}^k \to \mathbb{B}^n$ supposée linéaire.

**Définitions 7.7.**

(a) Une application $f : \mathbb{B}^k \longrightarrow \mathbb{B}^n$ est linéaire si elle vérifie

$$\forall u, v \in \mathbb{B}^k, \ f(u + v) = f(u) + f(v).$$

(b) L'image de $f$, notée $Imf$, est alors l'ensemble $Imf := \{y \in \mathbb{B}^n \mid \exists x \in \mathbb{B}^k : f(x) = y\}$.

(c) Le noyau de $f$, notéker $f$, est l'ensemble défini par $\ker f := \{x \in \mathbb{B}^k \mid f(x) = 0\}$.

**Proposition 7.8.** *Soit $f : \mathbb{B}^k \longrightarrow \mathbb{B}^n$ une application linéaire.*

*(a) L'image de $f$ est un sous vectoriel de $\mathbb{B}^n$.*

*(b) Le noyau de $f$ est un sous espace vectoriel de $\mathbb{B}^k$.*

*(c) $f$ est injective si et seulement si* $\ker f = \{0\}$.

**Remarques 7.9.** (a) La fonction d'encodage doit être injective ce que l'on décèle rapidement grâce au ($b$) de la proposition ci dessus.

(b) L'image $e(\mathbb{B}^k)$ étant alors effectivement un sous espace linéaire.

(c) La matrice $G$ est alors en fait la transposée de la matrice associée à $e$ dans les bases canoniques de $\mathbb{B}^k$ et $\mathbb{B}^n$.

**Exemples 7.10.** 1. 1) On donne la matrice suivante :

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

Cette matrice est-elle la matrice génératrice d'un code linéaire ?

Quelle est la longueur des mots codes ?

Quelle est la dimension du code ?

Combien y a t il de mots code ?

Donner une autre matrice génératrice du même code.

2) Considérons le codage $e$ défini par le tableau suivant :

| x | e(x) |
|---|---|
| 00 | 0000 |
| 01 | 0111 |
| 10 | 1001 |
| 11 | 1110 |

Quelle est la matrice de l'application $e$ (dans les bases canoniques) ?

Donner une matrice génératrice de ce codage.

Le poids minimum est 2 c'est un code 1 donc 1-détecteur.

**Définition 7.11.** On désigne par $p_1$ et $p_2$ les projections
$p_1 : \mathbb{B}^n \longrightarrow \mathbb{B}^n$ telle que $p_1(b_1 b_2 \dots b_n) = b_1 b_2 \dots b_k$. un codage est dit *systématique* s'il procède par adjonction de bits c'est à dire si, dans un certaine base, on a $p_1 \circ e = Id_{\mathbb{B}^k}$.

Le codage de l'exemple précédent est systématique.
Lorsqu'un codage linéaire $e$ est systématique alors, dans une certaine base, les $k$ premiers bits correspondent aux mots de départ. La matrice associée à $e$ est alors de la forme :

$$\begin{pmatrix} Id_k \\ K \end{pmatrix}$$

où $K$ est une matrice $(n-k)\times k$ qui correspond à la redondance. Du point de vue "matrice génératrice" il existe une base du code qui donne une matrice génératrice de la forme :

$$G = (Id_k \,|\, L)$$

Où $L$ est une matrice de type $k \times (n-k)$.

**Exemples 7.12.** 1) Dans l'exemple 7.10 2), la matrice $K$ est égale à

$$\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$$

2) Considérons le codage correspondant à l'ajout d'un bit de parité. On a donc $e : \mathbb{B}^k \longrightarrow \mathbb{B}^{k+1} : b_1b_2\dots b_k \mapsto b_1b_2\dots b_{k+1}$ où $b_{k+1}$ est le bit de parité égal à $b_{k+1} = \sum_{i=1}^{k} b_i$. La matrice correspondant à $e$ est donc

$$\begin{pmatrix} Id_k \\ K \end{pmatrix}$$

Où $K$ est la matrice ligne $11\dots1$ de longueur $k$.

Bien sur un code systématique est très facile à décoder. On a donc intérêt a essayer de s'y ramener. Pour cela on peut essayer de modifier la matrice génératrice du code. Remarquons à nouveau qu'il y a de nombreuses matrices génératrices pour un même code. Quelles sont les opérations que l'on peut faire subir a un matrice génératrice et encore obtenir une matrice génératrice ? Cela revient à demander quelles sont les opérations que l'on peut faire subir à une base et encore obtenir une base...Il s'agit donc des opérations de changements de base. Si $G \in M_{k\times n}(\mathbb{N})$ est une matrice génératrice du code les autres matrices génératrices du même code sont obtenues en multipliant à gauche la matrice $G$ par un matrice inversible de type $k \times k$. En particulier on peut permuter les lignes d'une matrice génératrice ajouter une ligne. ces deux opérations fournissent une nouvelle matrice génératrice du même code. Ces opérations ne garantissent pas que l'on puisse obtenir une matrice génératrice qui fasse de notre code un code systématique.

**Définition 7.13.** Deux codes linéaires $C$ et $C'$ de longueur $n$ sont équivalents s'il diffèrent uniquement par l'ordre des symboles des mots code. Autrement dit, s'il existe une permutation $(p_1, p_2, \dots, p_n)$ de $1, 2, \dots, n$ telle que pour tout mot code $v_1v_2\dots v_n$ est un mot code de $C$ si et seulement si $v_{p_1}v_{p_2}\dots v_{p_n}$ est un mot de $C'$.

**Proposition 7.14.** *Soit $C$ et $C'$ deux $(n,k)$-codes linéaires binaires ayant des matrices génératrices $G$ et $G'$ respectivement. Les codes $C$ et $c'$ sont équivalents si et seulement si $G'$ peut-être obtenue à partir de $G$ par une suite d'opérations des types suivants :*

*(a) Ajout d'une ligne à une autre*

*(b) Permutation de deux lignes*

*(c) Permutation de deux colonnes*

**Remarques 7.15.**

(a) Remarquons que les deux premières opérations (sur les lignes) ne changent pas le code. C'est à dire que ces opérations transforment une base de $C$ en une autre base de $C$.

(b) On peut montrer que tout code est équivalent à un code systématique.

(c) Lorsque l'on passe d'un code à un code systématique équivalent il ne faut pas oublier de revenir au code initial après avoir décodé ("facilement") le code systématique de repasser au code initiale.

**Définition 7.16.** Soit $C$ un $(n, k)$-code linéaire. Le code linéaire dual du code $C$ est défini par le sous espace :

$$C^{\perp} : \{y \in \mathbb{B}^n \,|\, x \cdot y = 0 \text{ pour tout x} \in \text{C}\}$$

Deux vecteurs $x, y \in \mathbb{B}^n$ sont orthogonaux si $x.y = 0$. Donc les vecteurs de $C^{\perp}$ sont ceux qui sont orthogonaux a tous les vecteurs de $C$. Si $C$ est un $(n, k)$-code alors $(C^{\perp})$ est un $(n, n-k)$-code. On a toujours $C = (C^{\perp})^{\perp}$. Un code est dit auto-dual si $C = C^{\perp}$.

**Exemple 7.17.** Le code dual du code $C = \{0000, 1111\}$ est

$$C^{\perp} := \{000, 1100, 1010, 1001, 0110, 0101, 0011, 1111\}$$

.

**Définition 7.18.** Soit $C$ un $(n, k)$-code. La matrice $H$ génératrice du code dual $C^{\perp}$ est appelée la matrice de contrôle du code $C$.

**Remarque 7.19.** *Puisque $(C^{\perp})$ est un $(n, n-k)$-code, la matrice de contrôle $H$ de $C$ est une matrice de type $(n-k) \times n$ dont les lignes sont linéairement indépendantes (base de $C^{\perp}$).*

**Théorème 7.20.** *Soient $G$ et $H$ les matrice génératrices et de contrôle d'un $(n, k)$-code linéaire $C$. Alors*

*(a)* $C = \{x \in \mathbb{B}^n \,|\, xH^t = 0 = Hx^t\}$.

*(b)* $GH^t = 0 = HG^t$

*(c) $d(C)$ est égal au nombre minimal de colonnes linéairement dépendantes dans $H$. En particulier on a toujours*

$$d(C) \leq n - k + 1.$$

*Cette borne est appelée la borne de Singleton.*

**Remarque 7.21.** *(a) Les codes qui sont tels que $d(C) = n-k+1$. sont appelés à distance de séparation maximale. our les paramètres $n, k$ donnés ils atteignent la meilleure distance minimale possible. On les désignent sous l'acronymes MDS (maximum distance separable).*

*(b) La réciproque du point b) du théorème 7.20 est également vraie : Si $G$ est une matrice $k \times n$, de rang $k$, à coefficients dans $\mathbb{B}$ et $H$ est une matrice $(n-k) \times n$ à coefficients dans  telle que $GH^t = 0$, alors $H$ est une matrice de parité du code $C$ ayant $G$ pour matrice génératrice.*

On a vu plus haut que tout code est équivalent à un code systématique. La matrice d'un code systématique est de la forme

$$G = (I_k \mid A)$$

où $I_k$ est la matrice identité de rang $k$et $A$ est une matrice $k \times (n-k)$ à coefficients dans $\mathbb{B}$. Si on pose $H = (A^t \mid I_{n-k})$, on vérifie facilement que $GH^t = A + A = 0$. La remarque ci-dessus montre que la matrice de contrôle associée au code engendré par $G$ est donc

$$H = (A^t \mid I_{n-k})$$

qui est donc une matrice de type $(n-k) \times n$ de rang $n-k$.

**Exemples 7.22.** 1) Trouver une matrice génératrice et une matrice de contrôle du code binaire $C := \{000, 111\}$

2) Trouver le code binaire linéaire dont la matrice de contrôle est :

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

Donner une matrice génératrice de $C$ et trouver le code dual. (deux méthodes possible pour le calcul de $G$ : directement en calculant le noyau de la matrice $H$, ou alors n transformant d'abord $H$ en une matrice "systématique"...)

3) Trouver la distance du $(n, n-1)$ code binaire $C$ ayant pour matrice de contrôle la matrice ligne $(11\ldots1)$.

# 8 Décodage des codes linéaires

### A) Tableau standard

**Définition 8.1.** Soit $C = e(\mathbb{B}^k) \subseteq \mathbb{B}^n$ un $(n,k)$-code linéaire. Pour $y \in \mathbb{B}^n$, on appelle $C$-classe de $y$ l'ensemble $y + C = \{y + c \mid c \in C\}$

**Proposition 8.2.** *Soit $y \in \mathbb{B}^n$ et $x \in C$ un point du code $C$ à distance minimale de $y$. Alors $y - x$ est de poids minimal dans la classe $y + C$.*

*Démonstration.* On a $\delta(y,x) \leq \delta(y,c)$ pour tout $c \in C$. Donc $w(y-x) = w(y-c)$ pour tout $c \in C$. □

En utilisant le principe du maximum de vraisemblance on peut donc dire que l'erreur $y - x$ est le vecteur de la classe de $y$ de poids minimal. Donc pour chaque classe $y + C$ de $C$ on déterminera (choix éventuel) un vecteur de poids minimal dans $y + C$. Soit $e$ un tel vecteur on décodera alors $y$ comme étant $y - e$. Le vecteur de poids minimal choisi dans la classe d'un élément $y \in \mathbb{B}^n$ sera appelé le *leader de la classe* déterminée par $y$.

Remarquons aussi : si $y_2 \in y_1 + C$ alors $y_1 + C \cap y_2 + C = \emptyset$ (justifier ceci). Sachant que $|C| = |e(\mathbb{B}^k)| = |\mathbb{B}^n| = 2^n$, on peut écrire

$$\mathbb{B}^n = \bigcup_{i=1}^{2^{n-k}} (y_i + C)$$

Pour chacune des $2^{n-k}$ classes on détermine un leader de classe et, pour chaque leader de classe on écrit tous les éléments de la classe. On a donc créé un tableau ou chaque élément de $\mathbb{B}^n$ apparaît une et une seule fois. pour décoder on regarde un mot reçu on regarde dans quel classe il se trouve et le leader de cette classe est alors l'erreur. On soustrait (=ajoute, car on ne considère que des codes binaires) cette erreur au mot reçu pour obtenir un mot code qui sera le mot décodé.

Le tableau standard a donc la forme suivante :

| | | | | | |
|---|---|---|---|---|---|
| $e_1 = 0 = c_1$ | $c_2$ | $\dots$ | $c_j$ | $\dots$ | $c_{2^k}$ |
| $e_2$ | $e_2 + c_2$ | $\dots$ | $e_2 + c_j$ | $\dots$ | $e_2 + c_{2^k}$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |
| $e_i$ | $e_i + c_2$ | $\dots$ | $e_i + c_j$ | $\vdots$ | $e_i + c_{2^k}$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |
| $e_{2^{n-k}}$ | $e_{2^{n-k}} + c_2$ | $\dots$ | $e_{2^{n-k}} + c_j$ | $\dots$ | $e_{2^{n-k}} + c_{2^k}$ |

**Exemple 8.3.** Donner un tableau standard pour le code binaire linéaire dont la matrice génératrice est donnée par :

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

et décoder le mot reçu 01111.

Les mots du code $C$ sont $C = \{00000\ 10101\ 01011\ 11110\}$. C'est un $(5, 2)$-code. Donc il y a $2^3 = 8$ classes distincts et donc le tableau standard aura 8 lignes. La distance minimale de $C$ est 3 et donc $t = 1$. Ce code détecte 2 erreurs et en corrige une.Les cinq vecteurs de poids un fournissent cinq classes (donc cinq lignes du tableau standard, pour ces cinq lignes là le leader de classe est unique ) et pour obtenir les trois lignes restantes on choisit u vecteur de poids deux qui n'est pas encore apparu dans les lignes précédentes. On obtient

| | | | |
|---|---|---|---|
| 00000 | 10101 | 01011 | 11110 |
| 10000 | 00101 | 11011 | 01110 |
| 01000 | 11101 | 00011 | 10110 |
| 00100 | 10001 | 01111 | 11010 |
| 00010 | 10111 | 01001 | 11100 |
| 00001 | 10100 | 01010 | 11111 |
| 11000 | 01101 | 10011 | 00110 |
| 10010 | 00111 | 11001 | 01100 |

Pour décoder le vecteur 01111 on trouve sa position dans la table. on remarque que 0111 se trouve dans la troisième colonne. Le mot code qui lui correspond est donc celui figurant en première ligne de cette troisième colonne. Le mot 01111 est donc décodé comme étant le mot code 01011.

**B) Syndrômes**

Pour un code de grande taille ($n >>$) le tableau standard n'est pas une bonne méthode de décodage.

**Définition 8.4.** Soit $C$ un $(n, k)$-code linéaire et $H$ sa matrice de contrôle. Pour un vecteur quelconque de $y \in \mathbb{B}^n$, on appelle *syndrôme* de $y$, noté $S(y)$, le vecteur

$$S(y) = yH^T$$

Où $H^T$ désigne la matrice transposée de la matrice de contrôle $H$. Remarquons que le syndrôme est défini par rapport à une matrice de contrôle, donc une autre matrice de contrôle donnera un autre syndrôme.
Remarquons que $S(y)$ est un vecteur ligne de longueur $n - k$. En outre $S(y) = 0$ si et seulement si $y \in C$. Notons également que $s(y) = s(y')$ si et seulement si $y - y' \in C$. Par conséquent deux vecteurs ont le même syndrôme si et seulement si ils appartiennent à la même classe définie par $C$. Il y a donc une correspondance bijective entre les syndrômes et les classes définies par $C$. La table des syndrômes se présente sous la forme de deux colonnes. la première comprend les leaders des classes et la seconde les syndrômes correspondants. Pour décoder un mot reçu on calcule son syndrôme, on trouve dans la table le leader de classe (c'est l'erreur) qui a le même syndrôme et on soustrait (= ajoute puisque l'on travaille en binaire) ce leader de classe au mot reçu pour obtenir le mot code qui est le plus proche (du mot reçu).

**Exemple 8.5.** On reprend le code de l'exemple 8.3. la matrice de contrôle correspondante est donnée par

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

On calcule alors $eH^T$ pour les leaders de classes. On obtient la table suivante :

| Leader | Syndrôme |
|---|---|
| 10000 | 101 |
| 01000 | 011 |
| 00100 | 100 |
| 00010 | 010 |
| 00001 | 001 |
| 11000 | 110 |
| 10010 | 111 |

Le syndrôme de $y = 11010$ est $yH^T = (11010)H^T = (100)$. Le leader de classe correspondant est obtenu grâce à la table des syndrômes ci-dessus. Le leader de classe dont le syndrôme vaut 100 est $e = 00100$. Le mot $y = 11010$ est donc décodé comme $x = y - e = 11110$.

**Exercices 8.6.**

1) Montrer qu'un code binaire linéaire ayant 20 mots code n'existe pas.

2) Montrer que le code $C = \{0000,\ 1010,\ 0101,\ 1111\}$ est linéaire et auto dual ($C = C^{\perp}$). Trouver sa matrice et sa matrice de contrôle.

3) Montrer que dans un code binaire linéaire soit tous les mots ont un poids paires soit la moitié d'entre eux ont un poids paires et l'autre moitié ont un poids impairs.

4 Si un bit de parité est ajouté à un code linéaire, le code obtenu est il encore linéaire ?

5) Trouver la matrice génératrice et la matrice de contrôle du code binaire ayant pour matrice génératrice la matrice $G$ suivante :

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

6) Soit $C$ le code binaire linéaire dont la matrice génératrice est :

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

Trouver une matrice génératrice sous forme standard pour ce même code (on n'utilise donc que des transformations sur le lignes).

6) Soit $C$ un $(7, 4)$-code binaire linéaire **parfait**. Calculer la distance de ce code. Supposons que ce code soit transmis via un canal dont la probabilité d'erreur sur chaque symbole est $p = 0,01$. Calculer la probabilité d'erreur sur un mot.

7) Soit $C$ le code binaire défini pa la matrice génératrice suivante :

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

a) Trouver une matrice de parité pour $C$
b) Montrer que $d(C) = 3$
c) Montrer que $C$ est parfait
d) Construire une table de syndrômes,
e) décoder les mots reçus 1110100 1010101 0011100 1000001.

# 9 Codes de Hamming

**Définition 9.1.** Soit $r \geq 1$ et $H$ la matrice $r \times 2^r - 1$ dont les colonnes sont les vecteurs non nuls de $\mathbb{B}^r$. Le code ayant $H$ pour matrice de contrôle est appelé code (binaire) de Hamming et noté $H(r, 2)$.

**Théorème 9.2.** *Les codes de Hamming $H(r, 2)$ sont des codes parfaits.*

*Démonstration.* C'est un exercice ! ! □

$H(r, 2)$ est un code de longueur $n = 2^r - 1$ et de dimension $n - r = 2^r - 1 - r$.

**Remarque 9.3.** *Remarquons que la définition ne précise pas l'ordre dans lequel les colonnes de $H$ correspondant aux vecteurs non nuls de $\mathbb{B}^r$ apparaissent. Bien sur ces codes sont tous équivalents. $r$ étant fixé on a donc en fait... $(2^r - 1)!$ codes équivalents.*

**Exemples 9.4.** a) Le code $H(2, 2)$ est un $(3, 1)$-code dont la matrice de contrôle est

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

En remarquant que cette matrice correspond à une matrice systématique on obtient facilement la matrice génératrice de ce code : $G = (111)$. C'est donc simplement le code de répétition binaire $H(2, 2) = \{000, 111\}$.

b) $H(3, 2)$ est un $(7, 4)$-code dont la matrice de contrôle est

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

Cette matrice est elle sous forme canonique (=systématique) ? Trouver la longueur, la dimension et la distance de ce code. Donner une matrice génératrice de ce code.

(c) vérifier que les deux codes de Hamming $H((2,2)$ et $H(3,2)$ sont parfaits.

.

**Remarque 9.5.** *On peut montrer que tout code binaire linéaire* 1*-correcteur parfait est un code de Hamming*

# 10 Codes cycliques

Les codes cycliques sont importants car ils ne nécessitent que très peu d'information pour être définis. Ils peuvent être très facilement implémentés grâce au registre à décalages. Beaucoup de codes importants en pratique sont des codes cycliques (les codes de Hamming binaires, les codes BCH, Reed-Solomon, Résidus quadratiques, Kerdock...)

**Définition 10.1.** Un code linéaire est dit cyclique si lorsqu'on permute cycliquement les lettres d'un mot code on obtient encore un mot code. Autrement dit un code linéaire $C$ de taille $n$ est cyclique si pour tout mot code $c_1 \dots c_n \in C$, le mot $c_n c_1 c_2 \dots c_{n-1} \in C$.

**Exemples 10.2.** 1) Un code répétitif est évidemment cyclique.

2) Un code à un bit de parité est aussi cyclique.

3) Le code binaire $C = \{0000, 1001, 0110, 1111\}$ n'est pax cyclique. Mais il est équivalent à un code cyclique (permuter les troisième et quatrième coordonnées)

En pratique les mots codes d'un code cyclique (binaire) de longueur $n$ sont souvent représentés sous la forme $c_0 c_1 \dots c_{n-1} \in \mathbb{B}^n$. A un tel mot code on associe le polynôme $c_0 + c_1 x + c_2 x^2 + \dots c_{n-1} x^{n-1} \in \mathbb{B}[x]_n := \{f(x) \in \mathbb{B}[x] \mid \deg f(x) < n\}$. Réciproquement à tout polynôme de $[x]_n$ (donc de degré $\leq n-1$) correspond un mot de $\mathbb{B}^n$. Si on multiplie le polynôme $c_0 + c_1 x + c_2 x^2 + \dots c_{n-1} x^{n-1} \in \mathbb{B}[x]$ par $x$ on obtient $c_0 x + c_1 x^2 + \dots + c_{n-1} x^n$ et en considérant alors le reste de la division euclidienne par $x^n - 1$ de ce dernier polynôme on obtient $c_n + c_0 x + c_1 x^2 + \dots + c_{n-1} x^{n-1}$ ce dernier polynôme correspond au mot $c_n c_1 c_2 \dots c_{n-1}$ On peut donc traduire la cyclicité d'un code $C$ en regardant l'ensemble $C(x)$ des polynômes associés aux mots du code $C$ et en demandant que cet ensemble $C(x)$ soit fermé pour les deux opérations suivantes :

- Si $c_1(x), c_2(x) \in C(x) \subset \mathbb{B}[x]$ alors $c_1(x) + c_2(x) \in C(x)$ (linéarité).
- Si $c(x) \in C(x)$ alors le reste de la division de $c(x)x$ par $x^n - 1$ est encore dans $C(x)$ (cyclicité).

**Remarque 10.3.** *Ce qui précède se résume en disant que l'image de* $C(x)$ *dans l'anneau* $A_n := \frac{\mathbb{B}[x]}{x^n - 1}$ *est un idéal de* $A_n$*. L'anneau* $A_n$ *est principal c'est à dire que tout idéal de* $A_n$ *est engendré par un unique élément. Autrement dit les éléments d'un idéal sont multiples d'un même élément. En outre cet élément est un diviseur de* $x^n - 1$*.*

**Théorème 10.4.** *Soit* $C$ *un code binaire cyclique de longueur* $n$*. Il existe un diviseur* $g(x) \in mathbbB[x]_n$ *de* $x^n - 1$ *tel que les éléments de* $C(x)$ *(ensemble des polynômes associés aux mots de* $C$*) soient les restes des multiples de* $g(x)$ *divisés par* $x^n - 1$*.*

Dit autrement, ce théorème affirme qu'il existe un polynôme diviseur de $x^n-$ tel que les mots de $C$ correspondent aux restes de la division par $x^n-1$ des polynômes de la forme $g(x)f(x)$ où $f(x) \in \mathbb{B}[x]$. Le code $C$ est donc entièrement défini par la donnée de la longueur $n$ et par la donnée d'un diviseur $g(x)$ de $x^n-1$. Pour reprendre nos notations habituelles, si $C$ est un $(n,k)$-code cyclique alors le polynôme $g(x)$ est de degré $n-k$ et il existe un polynôme $h(x)$ de degré $k$ tel que $g(x)h(x) = x^n - 1$.

**Lemme 10.5.** *Si $h(x) = h_k x^k + h_{k-1}x^{k-1} + \dots h_1 x + h_0$ divise $x^n - 1$ dans $[x]$ alors il en est de même du polynôme $\overline{h} := h_k + h_{k-1}x^1 + \dots h_0 x^k$.*

**Exercices 10.6.** Montrer que por un polynôme $h(x)$ de degré $m$ on a $\overline{h} = x^m h(x^{-1})$. En déduire une preuve du lemme ci dessus et auussi le fait que $\overline{gh} = \overline{g}\overline{h}$.

**Définitions 10.7.** Soit $C$ un $(n,k)$-code cyclique.

(a) Le polynôme $g(x)$ s'appelle le *polynôme générateur* du code cyclique $C$.

(b) Le polynôme $h(x)$ de degré $k$ tel que $g(x)h(x) = x^n - 1$ est appelé *polynôme de contrôle* de $C$.

(c) Le polynôme $\overline{h}$ associé au polynôme de contrôle $h$ est appelé le *polynôme réciproque* de $h$.

**Théorème 10.8.** *Soit $C$ un $(n,k)$-code cyclique. Alors*

*(a) le polynôme générateur est de degré $n-k$ et si $g(x) = g_0 + g_1 x + g_2 x^2 + \dots + g_{n-k}x^{n-k}$ est ce polynôme générateur alors une matrice génératrice de $C$ est*

$$G = \begin{pmatrix} g_0 & g_1 & g_2 & \cdots & g_{n-k} & 0 & 0 & 0 & \cdots & 0 \\ 0 & g_0 & g_1 & g_2 & \cdots & g_{n-k} & 0 & 0 & \cdots & 0 \\ 0 & 0 & g_0 & g_1 & g_2 & \cdots & g_{n-k} & 0 & \cdots & 0 \\ \vdots & & & & & & & & & \vdots \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & g_0 & \cdots & g_{n-k} \end{pmatrix}.$$

*(b) Si le polynôme de contrôle est $h(x) = h_0 + h_1 x + \dots + h_k x^k$, alors une matrice de contrôle du code $C$ est*

$$H = \begin{pmatrix} h_k & h_{k-1} & \cdots & h_0 & 0 & \cdots & 0 \\ 0 & h_k & \cdots & h_1 & h_0 & \cdots & 0 \\ \vdots & & & & & & \vdots \\ 0 & 0 & \cdots & 0 & h_k & \cdots & h_0 \end{pmatrix}$$

*De plus $C^\perp$ est le code cyclique engendré par le polynôme réciproque $\overline{h}(x) = h_k + h_{k-1}x + \dots + h_0 x^k$ de $h(x)$.*

**Exercices 10.9.** 1) Montrer que si $C_1$ et $C_2$ sont deux codes cycliques de longueur $n$ alors $C_1 \cap C_2$ est aussi un code cyclique. Si $g_1(x)$ et $g_2(x)$ sont le générateurs respectifs de ces deux codes, quel est le polynôme générateur du code $C_1 \cap C_2$ ?

2) Même question que ci-dessus pour la somme $C_1 + C_2 := \{x + y \mid x \in C_1,\ y \in C_2\}$.

3) On considère le polynôme $g(x) := 1 + x + x^3 \in \mathbb{B}[x]$. Trouver le polynôme $h(x)$ tel que $g(x)h(x) = x^7 - 1$. Construire la matrice génératrice du $(7, 4)$-code engendré par $g(x)$. Donner le polynôme de contrôle, son polynôme réciproque et une matrice de contrôle.

4) Montrer que le code de Hamming $H(3, 2)$ est cyclique. (En fait tous les codes de Hamming binaires sont cycliques).

5) On considère la factorisation $x^{23} - 1 = (x - 1)g_1(x)g_2(x)$ où $g_1(x) = x^1 1 + x^9 + x^7 + x^6 + x^5 + x + 1$ et $g_2(x) = x^1 1 + x^1 0 + x^6 + x^5 + x^4 + x^2 + 1$ Montrer que les polynômes $g_1$ et $g_2$ sont réciproques. Montrer que $c_1^{\perp}$ est le code cyclique engendré par $(x-1)g_1$. Montrer que la distance minimale du code $C_1$ est d'au plus 7 (remarquer qu le poids du mot code associé à $g_1$... En fait les codes $C_1$ et $C_2$ sont équivalents et ont une distance minimale de 7. Ce sont des codes 3 correcteurs. Ils sont appelés les *codes de Golay*

# 11 Gap et Guava

Notations : $GF(2)$ utilisé par Gap est notre corps à deux éléments $\mathbb{B}$.
Guava peut créé des codes répétitifs. Par exemple la commande
*gap* > C := RepetitionCode(5, GF(2)) ; ;
Qui crée un code répétitif de longueur 5. On peut alors demander la liste des mots code :
*gap* > cw := AsSortedList(C) ;
ce qui donne : [[00000], [11111]]
le poids d'un mot code est obtenu par la fonction WeightCodeword(C) : *gap* > WeightCodeword(Codeword("1011001")) ; 4
La distance (de Hamming) entre deux mots codes :
*gap* > a := Codeword("01111", GF(2)) ; ; *gap* > b := Codeword("11000",GF(2)) ; ; *gap* > DistanceCodeword(a,b) ; 4

**Exemple 11.1.** 1.2.8. soit $C = \{0000; 1100; 0011; 1111\}$. Alors $C$ est un $(4; 4; 2)$-code binaire. On peut utiliser la fonction *ElementsCode* pour construire un code éléments par éléments. Par exemple :
*gap* > C := ElementsCode(["0000", "1100", "0011", "1111"], GF(2)) ; ; construit le code défini ci dessus.

On peut obtenir la longueur, la taille (=nombre d'éléments) et la distance d'un code quelconque via les fonctions ci dessous : *gap* > n := WordLength(C) ; 4
*gap* > M := Size(C) ; 4
*gap* > d := MinimumDistance(C) ; 2
On peut utiliser Gap pour vérifier si un code est parfait. La fonction *IsPerfectCode* renvoie "true" si le code est parfait "false" si le code n'est pas parfait.
*gap* > C := RepetitionCode(5, GF(2)) ; ;

*gap* > IsPerfectCode(C) ; true
On peut utiliser Gap pour vérifier si un code est linéaire avec la fonction *IsLinearCode* par exemple :
*gap* > C := ElementsCode(["000","100", "001"], GF(2)) ;
*gap* > IsLinearCode(C) ;
et la réponse sera "true" si $C$ est linéaire et "false" si $C$ n'est pas linéaire.

On peut construire un code linéaire en donnant la matrice génératrice. par exemple si la matrice génératrice est :

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

une base du code est donc $[10001], [01010], [00111]$. On peut alors utiliser Gap :
*gap* > $C$ := GeneratorMatCode([[1,0,0,0,1],[0,1,0,1,0],[0,0,1,1,1]],GF(2)) ; ;

On peut retrouver la matrice génératrice de ce code en utilisant une combinaison des fonctions *Display* et *GeneratorMat*. Par exemple :
*gap* > Display(GeneratorMat(C)) ;
1 . . . 1
. 1 . 1 .
. . 1 1 1
Les points obtenus dans cette réponse sont évidemment des zéros.
On peut aussi utiliser Gap pour encoder. Par exemple si le message est [1, 1, 1] on utilise :
*gap* > m := Codeword("111", GF(2)) ;
*gap* > m * C ;
[ 1 1 1 0 0 ]

**Remarque 11.2.** *On ne doit utliser "GeneratorMat" pour calculer avec la matrice génératrice. On multiplie le message m avec "le code C".*

# Références


[D] M. Demazure, *Cours d'Algèbre*, Nouvelle bibliothèque Mathématique, Cassini, Paris 2008.

[DRTV] JG. Dumas, JL. Roch E. Tanier et S. Varette, *Théorie des Codes, Compression, Cryptage, Correction*, Dunod, Paris 2007.

[JN] S. K. Jain et S. R. Nagpaul : *Topics in apllied Algebra*, The Brooks/cole Series in Advanced Mathematics Thompson brooks/Cole, USA, 2005.