Chapitre IX

Codes linéaires et de Hamming

Introduction

Dans cette partie, on introduit une nouvelle classe de codes.
Ce type de code a pour but lors de la transmission d'un message à travers un canal de communication :

- de détecter ou de corriger les erreurs de communication dûes à la transmission
- de vérifier si un message transmis n'a pas été altéré par un acteur extérieur

Dans ce premier cours, nous commençons par formaliser le problème, et nous présentons une première classe de code : les codes linéaires.

1 Modélisation

1.1 Canal

Un canal est la modélisation théorique de tout moyen physique permettant de transmettre une information entre une émetteur et un récepteur.
Notations: on note :

- $\Sigma = \{s_1, \ldots, s_n\}$ l'alphabet contenant les symboles composant le message que l'on veut transmettre.
- $\Sigma' = \left\{s'_1, \ldots, s'_p\right\}$ l'alphabet contenant les symboles reçus.
- X une v.a. qui suit la loi des symboles de la source (ceux émis).
- Y une v.a. qui suit la loi des symboles après transmission (ceux reçus).

Définition 98 (Canal)**.**
Un canal (de communication) Γ transmettant des symboles issus d'un alphabet Σ est une matrice Γ de taille $\#\Sigma \times \#\Sigma'$ dont les éléments sont :

$$\Gamma_{i,j} = \Pr[Y = s'_j \mid X = s_i]$$

Ce type de canal est appelé un canal discret sans mémoire stationnaire :

- discret = évènements discrets,
- sans mémoire = le symbole reçu ne dépend que de celui qui a été envoyé, et non de l'ensemble des symboles précédents (causal), ou des suivants.
- stationnaire = la loi de Γ ne dépend pas de l'instant de la transmission, mais seulement des symboles transmis.

Remarques :

- Un canal bruité est un canal pour lequel $\exists i \neq j$ tel que $\Gamma_{i,j} \neq 0$ (un symbole reçu peut être différent d'une symbole transmis).
- Chaque ligne de cette matrice est la loi conditionnelle $\Pr[Y|X = s_i]$. En conséquence, $\sum_j \Gamma_{i,j} = 1$.
- Un canal est dit symétrique si sa matrice Γ est symétrique (i.e. $\Gamma = \Gamma^T$) ; autrement dit si $\Pr[Y = s'_j \mid X = s_i] = \Pr[Y = s'_i \mid X = s_j]$ (la probabilité que le symbole s_i soit transformé en symbole s_j est la même que la probabilité que le symbole s_j soit transformé en symbole s_i).

Définition 99 (Canal binaire symétrique (CBS))**.**
Un canal binaire symétrique correspond à une matrice de la forme :

$$\Gamma = \begin{bmatrix} \Gamma_{00} & \Gamma_{01} \\ \Gamma_{10} & \Gamma_{11} \end{bmatrix} = \begin{bmatrix} 1-e & e \\ e & 1-e \end{bmatrix}$$

où e est la probabilité de permutation d'un bit.

Un CBS correspond à une transmission binaire où la probabilité qu'un bit soit transmis sans erreur est $1 - e$, et la probabilité de permutation qu'un bit est e.

1.2 Transmission

Théorème 58 (Transmission d'une source S à travers un canal)**.**
Soient P le vecteur constitué par la loi marginale X des symboles de la source S, Q est le vecteur constitué par la loi marginale Y des symboles après transmission de la source S à travers le canal Γ.
Alors $Q = \Gamma.P$.

Démonstration:

Le calcul de $\Pr[Y = s'_j]$ s'effectue en multipliant la $i^{\text{ème}}$ ligne de la matrice par P et s'écrit $\sum_{s_i \in \Sigma} \Pr[Y = s'_j \mid X = s_i].\Pr[X = s_i]$

Or, ceci est la définition de $\Pr[Y = s'_j]$. □

Exemple

Soit le CBS tel que $e = 0.1$ et $P = \begin{bmatrix}0.7 & 0.3\end{bmatrix}^t$.
$Q = \Gamma.P = \begin{bmatrix}0.9 & 0.1\\ 0.1 & 0.9\end{bmatrix} \cdot \begin{bmatrix}0.7\\ 0.3\end{bmatrix} = \begin{bmatrix}0.66\\ 0.34\end{bmatrix}$
Donc $\Pr[Y = 0] = 0.66$ et $\Pr[Y = 1] = 0.34$.

1.3 Entropie

Définition 100 (Entropie conditionnelle d'une source à travers un canal)**.**
$H(\Gamma; X) = H(X|Y) = H(X, Y) - H(Y)$

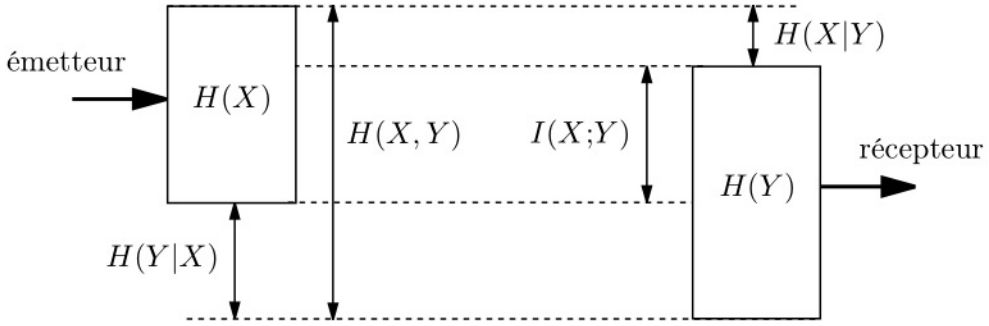
Par propriété de l'entropie conditionnelle $H(\Gamma; X) \leq H(X)$ avec égalité tient si X et Y sont des distributions indépendantes.
L'information mutuelle entre la source X et sa sortie Y du canal Γ est définie par :
$I(\Gamma; X) = H(X) - H(\Gamma; X) = H(X) - H(X|Y) = H(Y) - H(Y|X)$.

Exemple :

Dans le cas du CBS précédent, on $\Pr[Y = j|X = i] = \Gamma_{ij}$, $\Pr[X]$ par la loi marginale P et $\Pr[Y]$ par la loi marginale $Q = \Gamma.P = \begin{bmatrix}0.66 & 0.34\end{bmatrix}^t$.
$H(Y) = -\sum_i \Pr[Y = i] \log_2 \Pr[Y = i]$
$= 0.9248$ bits.
$H(Y|X) = -\sum_i \Pr[X = i] \sum_j \Pr[Y = j|X = i] \log_2 \Pr[Y = j|X = i]$
$= 0.4690$ bits.
D'où $I(\Gamma; X) = H(Y) - H(Y|X) = 0.4558$ bits.

On peut effectuer l'interprétation suivante :

- $H(X)$ est la quantité d'information de la source.
- $H(X|Y)$ est la quantité d'information permettant de lever l'ambiguité sur l'information transmise.
- $I(X; Y)$ est la quantité d'information transmise par le canal.



Autrement dit, une partie de l'information est perdue lors de la transmission.

1.4 Capacité d'un canal

Définition 101 (Capacité d'un canal)**.**
La capacité d'un canal est : $\gamma = \max_X I(\Gamma; X)$
à savoir la loi X qui maximise l'information mutuelle.

Remarques :

- γ est la capacité théorique du canal. Elle représente la meilleure utilisation possible du canal qui peut être faite.
- γ ne dépend pas de X mais seulement des caractéristiques du canal (Γ dans notre modélisation).

Exemple: (pour un CBS)
On note l'entropie du bruit $H_e = -(1-e)\log_2(1-e) - e\log_2 e$.

$$\begin{aligned} H(Y|X) &= -\sum_x p_X(x) \sum_y p_{Y|X}(x,y) \log_2 p_{Y|X}(x,y) \\ &= p.H_e + (1-p).H_e = H_e \end{aligned}$$

Donc, $I(\Gamma; X) = H(Y) - H(Y|X) = H(Y) - H_e$.

Or, $\gamma = \max_X I(\Gamma; X) = \max_X H(Y) - H_e$.

Comme Y dépend de X, le choix de X permet de maximiser Y, or pour un canal de 1 bit, $\max_{p\in[0,1]} H(p) = 1$. On en déduit, $\gamma = 1 - H_e$.

Conséquence : si $e = 0.1$, $\gamma = 1 - H(0.1) = 0.5310$ bit. Avec 10% d'erreur, la capacité du canal est presque divisée par 2.

Exercice 62: Canal bruité et entropie

On se place dans le cas d'un canal binaire symétrique.

1. On suppose une probabilité d'erreur $e = 20\%$.
   a) Donner la matrice Γ associée au CBS.
   b) Quel est le lien entre $\Pr[Y|X]$ et la matrice Γ ?
   c) Soit $\Pr[X = 0] = 0.4$ et $\Pr[X = 1] = 0.6$. Donner leurs probabilités après passage dans le canal de transmission.
   d) Calculer l'entropie du bruit.
   e) Donner l'entropie de la source à travers le canal.
   f) Donner la capacité du canal.
2. On se place maintenant dans le cas où la probabilité d'erreur est $e = 50\%$.
   a) Que se passe-t-il pour la matrice Γ ?
   b) Quelle est la conséquence sur la probabilité des bits à travers le canal de transmission ?
   c) Quelle est l'entropie de la source à travers le canal ?
   d) Donner la capacité du canal.

1.5 Théorème de Shannon

Dans le cours sur la compression, nous avions vu le premier théorème de Shannon qui disait que toute source pouvait être codée avec un nombre de bits par lettre aussi proche que l'on voulait de son entropie, et qu'il n'était pas possible de faire mieux.
Le second théorème de Shannon concerne le codage d'un canal.

Théorème 59 (second théorème de Shannon)**.**

- *Il est possible de transmettre une information de manière fiable (= en utilisant un code correcteur d'erreur) avec un taux de tranmission aussi proche que l'on veut de la capacité du canal.*
- *Il n'est pas possible de faire mieux.*

Comme pour son premier théorème, Shannon n'explique pas comment construire le code correcteur d'erreur permettant d'atteindre la capacité du canal, il démontre juste l'existence d'un tel code, et que la capacité du canal est la limite théorique du taux de transmission.

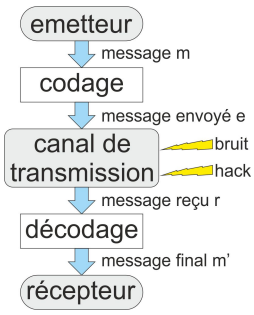
2 Codage

2.1 Définition

La présence d'erreur implique qu'il va être nécessaire transformer le message (= le coder) afin d'être en mesure de :

- soit détecter que le message a été modifié,
- soit corriger les erreurs dans le message.

On utilise le modèle de communication suivant :



- L'émetteur veut émettre un message m.
- Il code le message m en un message e.
- Il envoie le message e dans le canal de communication.
- Lors de sa transmission, le message e est bruité, ou subit une altération lors d'une interception malveillante.
- Le message reçu est r.
- r est alors décodé pour obtenir le message final m'.

Le message final m' peut :

- soit avoir été transmis sans erreur détectée, ou transmis avec erreur mais corrigé (*i.e.* lors du décodage),
  $\Rightarrow$ vrai positif
- soit avoir été transmis avec trop d'erreurs pour pouvoir être corrigé.
  Dans ce cas, le message est rejeté, et doit être retransmis.
  $\Rightarrow$ vrai négatif
- soit avoir été transmis avec erreurs, mais le codage ne permet pas de détecter les erreurs de transmission.
  Dans ce cas, le message est décodé incorrectement mais est considéré comme correct.
  $\Rightarrow$ faux positif

Le faux positif est impossible à éviter : il y aura toujours des cas où le bruit (= l'aléa) modifiera le message envoyé d'une façon cohérente (= non détectable par le décodeur).

On reprend les notations du premier chapitre :

- Un alphabet Σ (typiquement $\mathbb{B} = [0, 1]$) est l'ensemble des symboles utilisés pour la construction des mots.
- Un mot de longueur n est suite de symboles de taille n. C'est un élément de Σ^n.
- Un message m est une suite de symboles de Σ ($\in \Sigma^*$).

Définition 102 (Codage)**.** Un codage est une application injective $\phi : \Sigma^* \to \Sigma^*$.

En reprenant les notations précédentes : $e = \phi(m)$.

Si on note M l'ensemble des messages valides, ϕ est injective signifie que :

- tout message m a une image unique par ϕ, et en conséquence, ϕ est inversible sur l'image de $\phi = \phi(M)$ (*i.e.* si $e = \phi(m)$ alors $m = \phi^{-1}(e)$).
- inversement, le message reçu r n'est pas nécessairement dans $\phi(M)$, et en conséquence, il peut ne pas être décodable (*i.e.* $\phi^{-1}(r)$ n'existe pas pour ce r).

Définition 103 (Codage par bloc)**.**
Le message $m \in \Sigma^*$ est découpé en bloc B_i de taille p (i.e. $m = B_0 \dots B_k$).
Le codage ϕ est un application injective de codage par bloc $\Sigma^p \to \Sigma^n$.
Le message e est transmis par bloc : $e = (e_0, \dots, e_k) = (\phi(B_0), \dots, \phi(B_k))$.

Cette transformation conduit nécessairement à l'augmentation de la taille du code ($n > p$) afin d'intégrer les informations nécessaires à la détection d'erreur et la correction.

Définition 104 (Mots de code)**.**

L'ensemble des éléments $C = \{\phi(m) \mid m \in \Sigma^p\}$ sont les mots de code de ϕ.

Notes :

- les mots de Σ^p sont les mots sources qui composent le message.
- C est un sous-espace de Σ^n.

Exemple : soit $\phi : \Sigma^2 \to \Sigma^3$, $\phi(b_0b_1) = b_0b_1b_2$ avec $b_2 = (b_0 + b_1) \bmod 2$.

- $\Sigma^2 = \{00, 01, 10, 11\}$.
- $C = \Phi(\Sigma^2) = \{000, 011, 101, 110\}$.

Définition 105 (Codage systématique)**.**
Un codage ϕ est dit systématique si $\forall x \in \Sigma^p$, le codage $\phi(x)$ peut s'écrire comme xy où x est le code source et y la partie redondante.

Note : L'écriture de $\phi(x)$ sous forme xy peut éventuellement être une permutation des symboles xy.

Exemple : le codage ϕ vu à l'exemple précédent est systématique. $\phi'(b_0b_1) = b_0b_2b_1$ et $\phi''(b_0b_1) = b_0b_0b_1b_1$ ne sont pas systématique mais peuvent être mis sous forme systématique.

Définition 106 (Code de taille (p, n))**.** Un code par bloc est dit de taille (p, n) si sa fonction de codage ϕ est définie de Σ^p dans Σ^n.

Notes :

- ne pas confondre code (= message codé), mot de code, mot de source, et codage (= action d'appliquer la fonction de codage sur le message).
- la fonction ϕ de l'exemple précédent produit un code $(2, 3)$.

2.2 Erreur

Hypothèse sur les erreurs :

- Les erreurs de transmissions changent les symboles (pas d'insertion, de suppression, de perte, ...),
  $e = (X_1X_2 \dots X_k)$ et $r = (Y_1Y_2 \dots Y_k)$
- Chaque symbole a la même probabilité d'être altéré lors de la transmission, $\forall i \neq j, \Pr[Y_i \neq X_i] = \Pr[Y_j \neq X_j]$.
- La probabilité d'erreur sur chaque symbole est indépendante. $\Pr[Y_1 \neq X_1, Y_2 \neq X_2 \dots Y_k \neq X_k] = \Pr[Y_1 \neq X_1].\Pr[Y_2 \neq X_2] \dots \Pr[Y_k \neq X_k]$

Ces hypothèses correspondent à des erreurs apparaissant de manière ponctuelle et aléatoire.

Remarque : dans le cas d'erreurs en rafale, le codage par bloc classique n'est plus efficace, et nécessite d'utiliser un entrelacement du message avant de le coder par bloc.

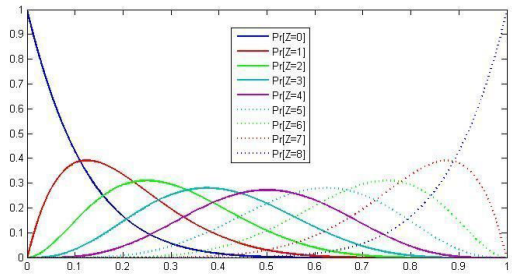
Sous ces hypothèses, pour un bloc de taille n et avec un taux d'erreur q (*i.e.* probabilité qu'un bit change d'état),

La loi associée Z au nombre d'erreurs est une loi binomiale :

$$\Pr[Z = k] = C_n^k.q^k.(1-q)^{n-k}$$

où $q^k.(1-q)^{n-k}$ est la probabilité d'avoir k erreurs sur le bloc de taille n, et C_n^k est le nombre de façons de placer k erreurs sur n bits.

Exemple :
Lois d'erreur pour un bloc de 8 bits



si $q = 0.01$, alors pour :
$n = 8$, $\Pr[Z = 0] = 0.9227$
$n = 16$, $\Pr[Z = 0] = 0.8515$
$n = 24$, $\Pr[Z = 0] = 0.7857$
$n = 32$, $\Pr[Z = 0] = 0.7250$
$n = 48$, $\Pr[Z = 0] = 0.6173$
$n = 64$, $\Pr[Z = 0] = 0.5256$

Hypothèses supplémentaires : q est suffisament petit de façon à ce que $\Pr[Z = 0] > \Pr[Z > 0]$ (i.e. il est plus probable qu'il n'y ait pas d'erreur de transmission).

Définition 107 (distance de Hamming de deux mots)**.**
Soit $w, w' \in \Sigma^p$, la distance de Hamming (notée d_H) entre deux mots d'un code est le nombre de position où w et w' ont des symboles différents :

$$d_H(w, w') = \#\left\{i \mid w_i \neq w'_i\right\}$$

La distance de Hamming peut également s'interpréter comme :

- le nombre de symboles à modifier pour transformer w en w' (et vice-versa),
- le nombre d'erreur à commettre sur w pour le confondre avec w' (et vice-versa)

Note :

La distance de Hammming a toutes les propriétés d'une distance, à savoir $d_H(x, y) \geq 0$ avec $d_H(x, y) = 0$ si $x = y$ (positivité), $d_H(x, y) = d_H(y, x)$ (symétrie), et $d_H(x, y) \leq d_H(x, z) + d_H(z, y)$ (inégalité triangulaire).

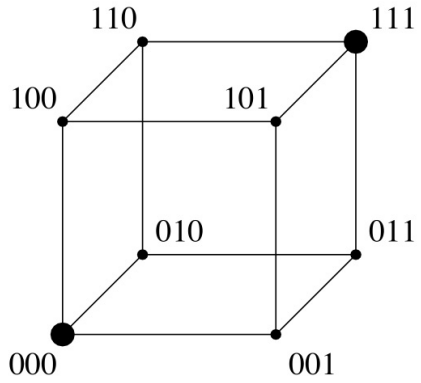
Exemple :
$d_H(010, 110) = 1$,
$d_H(1100, 0110) = 2$.

Définition 108 (distance de Hamming d'un code)**.**
La distance de Hamming d'un code C est la plus petite distance entre ses symboles, à savoir :

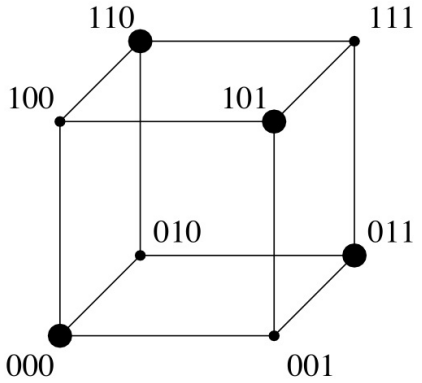
$$d_H(C) = \min_{(w,w') \in \Sigma^{n^2}, w \neq w'} d_H(w, w').$$

Exemple : on représente les codes de $\{0, 1\}^3$ comme les sommets d'un cube. La distance de Hamming peut être aussi vue comme le petit nombre d'arêtes reliant les mots du code (placés aux sommets).

$C_1 = \{000, 111\}$
$d_H(C_1) = 3$



$C_2 = \{000, 011, 101, 110\}$
$d_H(C_2) = 2$



Théorème 60 (Capacité de correction d'un code)**.**
Soit Σ *un code C de distance de Hamming D.*
Alors, si e est le code émis, et r le code reçu, C permet de :
• *détecter des erreurs si* $d_H(e, r) < D$,
• *corriger les erreurs si* $d_H(e, r) < D/2$.

A savoir, le code C permet de détecter jusqu'à $D - 1$ erreurs et corriger jusqu'à $\lfloor \frac{D-1}{2} \rfloor$ erreurs.

Démonstration:

- si $0 < d_H(e,r) < D$, alors comme $e \in C$, $\nexists e' \in C \mid r = e'$ (puisque la distance minimale entre deux mots du code est D. En conséquence, il y a eu une erreur de transmission.
- si $0 < d_H(e,r) < D/2$.
  Par définition, pour tout $e' \in C \setminus \{e\}$, $d(e,e') \geq D$.
  $d_H(e',e) \leq d_H(e',r) + d_H(r,e)$, donc $D \leq d_H(e',r) + d_H(r,e)$.
  Et comme $d_H(e,r) < D/2$, $D - d_H(e',r) = D/2 \leq d_H(r,e)$.
  En conséquence, $d_H(e,r) < D/2$ et $D/2 \leq d_H(r,e)$ implique que e est le mot le plus proche de r que tout autre mot de l'alphabet.
  r peut être donc corrigé en e.

□

Il y a une deuxième façon de voir le problème détection/correction d'erreurs.

Définition 109 (Sphère de Hamming $S_t(x)$)**.**
Une sphère de Hamming $S_t(x)$ de rayon t et centrée en un mot x de Σ^n est l'ensemble des mots situés à une distance inférieure à t de x.

$S_t(x) = \{y \in \Sigma^n \mid d_H(x,y) \leq t\}$

Alors avec cette définition,

- La capacité de détection d'un code C est le rayon de la plus grande sphère de Hamming contenant un seul mot du code.
- La capacité de correction d'un code C est le plus grand rayon r tel qu'aucune des sphères de Hamming de ce rayon et centré en tous les mots de code C ne s'intersectent.

Il s'agit donc d'un problème qui peut s'apparenter (lorsqu'il est vu sous cette forme) au problème du "sphere packing" (*i.e.* à savoir dans un espace donné, comment placer le maximum de sphère d'un diamètre donné sans que celles-ci s'intersectent).

Définition 110 (code $[n, p, d_{\min}]$)**.**
Un code C est dit $[n, p, d_{\min}]$ si :

- c'est un code (p, n) (= sa fonction de codage ϕ est définie de Σ^p dans Σ^n),
- $d_{\min} = d(C)$ (= sa distance de Hamming est $d_{\min}$).

Définition 111 (taux d'un code)**.**
Le taux d'un code (ou rendement) d'un code (p, n) ou $[n, p, d_{\min}]$ est n/p

Remarque : le rendement est le nombre de bits d'information (*i.e.* ceux du message) par bits codés (*i.e.* ceux du message codé).

En conséquence, la performance d'un (p, n) code C détecteur/correcteur d'erreur se mesure à :

- La distance de Hamming du code qui déterminé les capacités du code (voir théorème précédent),
- L'augmentation de la taille d'un message n/p lors de son codage (= son taux).

Le choix de p et D dépend alors du contexte :

- taux d'erreur faible + reexpédition peu coûteuse : détection erreur + reexpédition.
  code à choisir : à p et n fixés, celui qui maximise D.
- taux d'erreur élevé + reexpédition coûteuse : détection erreur + correction.
  code à choisir : à p et D fixés, celui qui minimise n.

Exercice 63: Distance de Hamming

Soit le code $C = \{000000, 000111, 111000, 111111\}$.

1. Donner la distance de Hamming du code.
2. Donner un exemple démontrant la capacité de détection d'erreur.
3. Donner la capacité de correction du code.
4. Donner un exemple démontrant la capacité de détection d'erreur.
5. Donner les caractérisques de ce code.
6. Donner le taux de code.

On a vu que par le théorème de Bayes :

$$\Pr[w|b_1 \dots b_n] = \frac{\Pr[w].\Pr[b_1 \dots b_n|w]}{\Pr[b_1 \dots b_n]}$$

où w est le bloc transmis (avant codage) et $b_1 \dots b_n$ sont les bits reçus.

Le décodage optimal consiste à choisir w tel que :

$$w_{\text{opt}} = \arg\max_{w \in \Sigma} \Pr[w|b_1 \dots b_n]$$

Note : si tous les symboles de l'alphabet Σ sont équiprobables, alors ce choix est équivalent au maximum de vraisemblance $\Pr[b_1 \dots b_n|w]$.

3 Premiers exemples

3.1 Répétition

Définition 112 (codage binaire par répétition $(1, n)$)**.**

Un codage par répétition consiste à répéter chaque bit un nombre n impair de fois. A savoir, soit $\phi : \Sigma \rightarrow \Sigma^n$
$$x \mapsto x^n$$

Exemple : le chaine 1011 est décomposée en blocs de 1 bit, puis codée comme $\phi(1)\phi(0)\phi(1)\phi(1) = 111000111111$.

Comment décoder ?

Sous les hypothèses déjà énoncées, chaque bit a une probabilité < 0.5 d'avoir une erreur, et cette probabilité est indépendante pour chaque bit. On rappelle que $\Pr[Z = k]$ est la probabilité d'avoir k erreurs sur un bloc de n bits.

En conséquence, si l'on reçoit un paquet de n bits, et que q de ces bits sont égaux à 0, alors il y a deux cas :

- si $q > n - q$, alors $\Pr[Z = r] < \Pr[Z = n - r]$, et on décide $e = 0^n$.
- si $q < n - q$, alors $\Pr[Z = r] > \Pr[Z = n - r]$, et on décide $e = 1^n$.

Clairement, ce choix est équivalent à un vote majoritaire en faveur du bit le plus fréquent.

Quels sont les capacités d'un tel code ?

Le code C est donc constitué de deux symboles 0^n et 1^n, $d_H(C) = d_H(0^n, 1^n) = n$.

D'après le théorème de correction, ce code détecte les erreurs inférieures à n et corrige les erreurs $n < 2$ (ce que nous confirme le résultat du décodage précédent).

La probabilité pour un bit d'être décodé incorrectement est donc :

$$Pr[Z > n/2] = \sum_{i=\lceil n/2 \rceil}^{n} \binom{n}{i} q^i.(1-q)^{n-i} \simeq \Phi\left(\sqrt{n}\frac{1/2-p}{\sqrt{p(1-p)}}\right) \text{ quand } n \rightarrow \infty$$

par le théorème de Moivre-Laplace où Φ est la loi normale centrée réduite. En conséquence, lorsque $n \rightarrow \infty$, $Pr[Z > n/2] \rightarrow 0$.

Donc, augmenter le nombre de répétitions :

- permet de faire tendre la probabilité d'erreur vers 0,
- mais fait aussi tendre le rendement du code vers 0 (= proportion d'informations utile)

Plus ce code est capable de corriger d'erreur, plus il est mauvais.

Par le second théorème de Shannon, on sait qu'il est donc possible de trouver

mieux.

Exercice 64: Code binaire à répétition

On considère un code par répétition $(1, 5)$.

1. Donner le codage par répétition de la chaine 11010.
2. Donner la distance de Hamming de ce code.
3. Donner la capacité de correction du code.
4. Donner le rendement de code.

3.2 Parité

Définition 113 (Parité)**.**
La parité p d'une suite de bits $\{b_i\}_{i=1\ldots n}$ est égale à $p = (\sum_i b_i) \bmod 2$.

Notes :

- si le contexte binaire est implicite, l'addition sur les bits s'effectue modulo 2.
- la parité est nulle si le nombre de bits non nuls est pair.
- autre interprétation : $p + \sum_i b_i = 0$.

Exemple: premier exemple

soit $\phi(b_1 \ldots b_n) = b_1 \ldots b_n p$ où $p = \sum_i b_i$.

alors le bit de parité p permet de détecter un nombre impair d'erreurs (y compris sur le bit de parité lui-même).

Prenons $S = \{00, 01, 10, 11\}$, $C = \{000, 011, 101, 110\}$ alors $d_H(S) = 1$ et $d_H(C) = 2$.

Idem avec n bits, donc ajouter un bit de parité permet d'augmenter la distance de Hamming de 1.

Problème : il ne permet pas de savoir quel bit a été modifié.

Plusieurs questions se posent alors si l'on veut corriger plus qu'une seule erreur :

- faut-il répéter le bit de parité ?
  exemple : $\phi(b_1 b_2) = b_1 b_2 pp$ où $p = b_1 + b_2$.
  Si b_1 ou b_2 est modifié, détection avec le bit de parité mais impossibilité de savoir lequel des deux a été modifié ($d_H(C) = 2$).
  Il faut donc utiliser des combinaisons linéaires de parité différentes.
- quelles combinaisons linéaires de parité ?
  exemple 1 : $\phi(b_1 b_2 b_3) = b_1 b_2 b_3 p_1 p_2$ avec $p_1 = b_1 + b_2$ et $p_2 = b_2 + b_3$.
  $d_H(C) = 2$, donc ce code n'est pas correcteur.
  exemple 2 : $\phi(b_1 b_2 b_3) = b_1 b_2 b_3 p_1 p_2 p_3$ avec $p_i = \sum_{j=1}^{i} b_j$.

$d_H(C) = 2$, donc ce code n'est pas correcteur.

exemple 3 : $\phi(b_1b_2b_3) = b_1b_2b_3p_1p_2p_3$ avec $p_i = \sum_{j\neq i} b_j$.

$d_H(C) = 3$, donc ce code corriger 1 bit.

comment corriger un tel code ?

Il est donc nécessaire de découvrir les règles qui permettent de construire des codes robustes, ainsi que les façons de corriger ces codes. Nous formalisons maintenant ce problème dans le cadre des codes linéaires.

4 Codes linéaires

4.1 Définition

Définition 114 (Code linéaire)**.**
Un code (p, n) est un code linéaire si ce code C peut s'écrire sous la forme :

$$C = \{y \mid y = x.G\}$$

où $x \in \Sigma^p$ est le bloc de taille p à coder, et G est une matrice de taille $n \times p$ appelée **matrice génératrice** du code C.

Notes :

- la fonction de codage ϕ s'écrit : $\phi(x) = x.G$.
- les vecteurs seront stockés sous forme de vecteur ligne, et pour les calculs matriciels, les multiplications s'effectueront à gauche.
- pour $\Sigma = [0; 1]$, les calculs sont menés modulo 2.
  De manière équivalente, l'addition binaire correspond à un **OU**-exclusif et la multiplication binaire correspond à un **ET** logique.

Exemple: $\phi : \Sigma^2 \to \Sigma^3$**,** $\phi(b_0b_1) = b_0b_1b_2$ **avec** $b_2 = b_0 + b_1$

ϕ est un code linéaire, et sa matrice de génération est $G = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$.

En effet, $\phi(b_0b_1) = \begin{bmatrix} b_0 & b_1 \end{bmatrix} . \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} b_0 & b_1 & b_0 + b_1 \end{bmatrix}$.

Définition 115 (Représentation systématique d'une matrice génératrice)**.**
Si le codage d'un code linéaire (p, n) est systématique, alors la matrice génératrice G d'un peut être mis sous la forme systématique suivante :

$$G_{\text{sys}} = \left[\; I_p \mid P \; \right]$$

où I_p est la matrice identité de taille $p \times p$, et P une matrice de parité de taille $p \times (n - p)$.

Exemple: En reprenant l'exemple précédent,

G est déjà sous forme systématique :

$$G = \left[\begin{array}{cc|c} 1 & 0 & 1 \\ 0 & 1 & 1 \end{array}\right] = \left[\begin{array}{c|c} I_2 & P \end{array}\right] \text{ où } P^T = \begin{bmatrix} 1 & 1 \end{bmatrix}.$$

Note : la matrice I_p permet d'assurer que $\phi(x)$ commence par x.

4.2 Matrice de contrôle

Comme ϕ est injective, et $\phi : \Sigma^p \to \Sigma^n$, on a $C = \phi(\Sigma^p) \subset \Sigma^n$, et C est un sous-espace vectoriel de dimension p (combinaison linéaire par G des vecteurs canoniques de la base de Σ^p).
En conséquence, il existe un espace dual $C^\perp$ à C.

Définition 116 (Matrice de contrôle)**.**
Soit un code C linéaire (p, n) de matrice génératrice G. Alors il existe une matrice H (dite de contrôle), telle que :

$G.H^t = 0_p$ où 0_p est une matrice de taille $p \times p$.
$\forall c \in C, c.H^t = 0^p$ où 0^p est un vecteur de p zéro.

Note : cette matrice est également appelée matrice de vérification de parité.

Théorème 61 (Calcul de la matrice de contrôle)**.**
Si G est la matrice génératrice d'un code linéaire (p, n) sous forme systématique, alors $H = \left[\begin{array}{c|c} P^t & I_{n-p} \end{array}\right]$.

Démonstration:
$G.H^t = \left[\begin{array}{c|c} I_p & P \end{array}\right].\left[\begin{array}{c|c} P^t & I_{n-p} \end{array}\right] = -P + P = 0.$ □

Exemple: code binaire linéaire $[4, 2, 2]$

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

Mise sous forme systématique : $G_{\text{sys}} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}$

La matrice de parité est : $P = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$

On en déduit la matrice de contrôle : $H = \left[\begin{array}{c|c} P^t & I_{n-p} \end{array}\right] = \left[\begin{array}{cc|cc} 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{array}\right]$

On a donc le codage suivant :

c	00	01	10	11
$\phi(c) = c.G$	0000	0111	1010	1101
$\phi(c).H^t$	00	00	00	00

Si on prend un mot qui n'est pas du code $x = 0101$, $x.H^t = 01 \neq 00$.

4.3 Poids de Hamming

Définition 117 (Poids de Hamming)**.**
Le poids de Hamming $w_H(x)$ d'un vecteur x de Σ^n est le nombre de composantes non nulles de x.

Note : $\forall x \in \Sigma^n, w_H(x) = d_H(x, 0^n)$.

Propriété 62 (Poids de Hamming pour les codes binaires linéaires)**.**

1. $d_H(x, y) = w_H(x + y)$.
2. *Le calcul de la distance de Hamming d'un code est équivalent au calcul du poids minimum de Hamming de tous les mots de code non nuls.*

Démonstration:

1. $d_H(x, y) = d_H(x + y, 0^n) = w_H(x + y)$ par linéarité de $x + y$.
2. $d_H(C) = \min\limits_{(u,v)\in C^2, u\neq v} d_H(u, v) = \min\limits_{(u,v)\in C^2, u\neq v} w_H(u + v) = \min\limits_{u\in C, u\neq 0} w_H(u)$ car C est un espace vectoriel.

□

4.4 Syndrome

Pour décoder un code linéaire, il faut être en mesure de déterminer, pour chaque bloc $r \in \Sigma^n$ reçu,

- si le message r a été modifié,
- s'il l'a été, quelle est le le message m envoyé le plus probable (*i.e.* le plus "proche" de $\phi(m)$).

Mais comment détecter si le message a été modifié ?

Définition 118 (Syndrome)**.**
Soit un code linaire C ayant pour matrice de contrôle H.
Un mot émis par le code C est transmis et reçu comme r.
Alors, le syndrome s de r est $r.H^t$.

On sait que pour tous les mots du code $e \in C, e.H^t = 0^p$. La détection d'erreur peut donc s'effectuer en utilisant la matrice de contrôle.
Soit $r = e + \epsilon$ le mot reçu (ϵ est l'erreur de transmission). Alors le syndrome de r est $s = r.H^t = (e + \epsilon).H^t = \epsilon.H^t$.

Donc le syndrome s d'un mot reçu est caractéristique de l'erreur de transmission :

- si $s = 0^p$, alors le mot est supposé avec été reçu sans erreur.
- si $s \neq 0^p$, alors le mot reçu contient (au moins) une erreur.

Comme s est le syndrome d'un code linéaire (p, n), $s \in \Sigma^p$. Or comme $r \in \Sigma^n$ et que $n > p$, plusieurs r différents peuvent avoir le même syndrome.
On fait alors l'hypothèse suivante :

Hypothèse 119 (Erreur d'un code linéaire)**.**
Le type d'erreur est celui d'un CBS pour d'erreur e, à savoir :
si $e = (e_1 \dots e_n)$ le mot émis et $r = (r_1 \dots r_n)$ le mot reçu, on a :

1. $\Pr[e = r] > \Pr[e \neq r]$.
2. $\forall i, \Pr[e_i = r_i] > \Pr[e_i \neq r_i]$.
3. $\Pr[d_H(e, r) = k] > \Pr[d_H(e, r) = k + 1]$.

Note : en conséquence, l'absence d'erreur (ou du nombre minimum d'erreur) est supposée toujours plus probable qu'un nombre d'erreurs plus élevé.

A cette fin, on construit un tableau dit "standard" de la manière suivante :

Définition 120 (Tableau standard associé à un (p, n) code C)**.**
Tableau construit de la manière suivante :

- en tête de la $i^{\text{ème}}$ colonne, les mots sources $m_i \in \Sigma^p$ et son codage $\phi(m_i) \in \Sigma^n$.
- en tête de la $j^{\text{ème}}$ ligne (au plus 2^{n-p} lignes, une par syndrome différent)
  - ⋄ les erreurs $\epsilon_j \in \Sigma^n$ triées par ordre croissant de poids $w_H(\epsilon_j)$,
  - ⋄ le syndrome de ϵ_j ($s_j = \epsilon_j.H^t$).
- à l'intersection de la $i^{\text{ème}}$ colonne et de la $j^{\text{ème}}$ ligne, la somme de $m_i + \epsilon_j$.

Exemple: Tableau standard du code binaire linéaire $[4, 2, 2]$

		00	01	10	11	c
ϵ_i	$\epsilon_i.H^t$	0000	0111	1010	1101	$\phi(c)$
0000	00	0000	0111	1010	1101	
1000	10	1000	1111	0010	0101	
0100	11	0100	0011	1110	1001	
0010	10	~~0010~~	~~0101~~	~~1000~~	~~1111~~	fait
0001	01	0001	0110	1011	1100	

fait = ces valeurs ont déjà été obtenue avec une autre erreur (ici $\epsilon = 1000$). On remarque que le syndrome est le même, et que l'on obtient les mêmes valeurs.

Analyse :

- Le code binaire linéaire $[4, 2, 2]$, donc sa distance de Hamming est $D = 2$. En conséquence, ce code à une capacité de détection de 1 erreur $(D - 1)$, et une capacité de correction de 0 $(< \lfloor (D - 1)/2 \rfloor)$.

- Chaque syndrome différent subdivise Σ^n en classe d'équivalence (nommée coset).
  noter que le syndrome que l'on avait déjà engendre les mêmes codes.
- Le représentant (chef) d'un coset est le vecteur ϵ_i qui le génère.
- La correction d'un mot reçu consiste à rechercher dans le tableau, et à prendre comme correction le code source associé à la colonne.
- Lorsque l'on est en dessous de la capacité de correction du code, le code source associé peut dépendre de l'ordre de choix du chef de coset.
  - ◇ 1000 et 0010 ont tous deux pour syndrome 10.
  - ◇ 1111 se décode comme 01 si le chef de coset est 1000 et comme 11 si le chef de coset est 0010.

  On dit alors qu'un tel code n'est pas corrigé équitablement.
- la classe est caractérisée par son syndrome, et par son chef de coset.
  Comme $r = m + \epsilon$, on a aussi $m = r - \epsilon = r + \epsilon$.

Pour décoder un (p, n)-code linéaire, si on reçoit r, il faut :

1. calculer le syndrome de r : $s = r.H^t$.
2. à partir de s, trouver le chef de coset ϵ_i associé.
3. calculer le code corrigé $\hat{r} = r - \epsilon_i$.
4. comme le code est systématique, le message est constitué des p premiers symboles de $\hat{r}$.

Exemple: Tableau standard du code binaire linéaire $[4, 2, 2]$
Le tableau standard réduit suivant suffit pour décoder :

syndrome	00	01	10	11
ϵ_i	0000	0001	1000	0100

Supposons $r = 1110$,

1. $s = r.\begin{bmatrix}1 & 1 & 1 & 0\\ 0 & 1 & 0 & 1\end{bmatrix}^t = 11.$
2. $s = 11 \Rightarrow \epsilon_i = 0100.$
3. $\hat{r} = r - \epsilon_i = 1110 - 0100 = 1010.$
4. $\hat{r} = \phi(c) = 1010 \Rightarrow c = 10.$

Il est également possible d'analyser le tableau standard en terme de distance :

- La colonne ϵ_i a été construite à partir de l'ensemble des erreurs de plus petits poids ayant un syndrome différent.
- En conséquence, la $i^{\text{ème}}$ colonne contient l'ensemble des mots transmis les plus proches d'un mot du code.
- Cette $i^{\text{ème}}$ colonne est la région de décodage du mot de code c_i (à savoir l'ensemble des mots qui sont décodés comme c_i).

Plus formellement,

Définition 121 (Région de décodage)**.**
La région de décodage R_i autour d'un code c_i est définie comme :

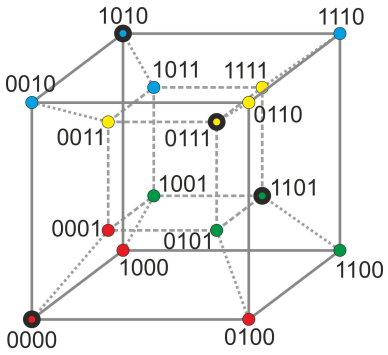
$$R_i = \left\{ c_i + \epsilon_j \mid \epsilon_j \text{ est l'ensemble des chefs de coset} \right\}$$

Avec cette définition, et un $[n, p, d]$ code binaire linéaire C :

- par construction, $\#R_i = 2^{n-p}$.
- C a une capacité de correction de $t = \lfloor (d-1)/2 \rfloor$, et en conséquence, chaque sphère de Hamming $S_t(c_j)$ est incluse dans R_i (*i.e.* $\forall i, S_t(c_j) \subseteq R_i$).

Exemple: Régions associées au code binaire linéaire $[4, 2, 2]$

		00	01	10	11	c_i
ϵ_i	$\epsilon_i.H^t$	0000	0111	1010	1101	$\phi(c_i)$
0000	00	0000	0111	1010	1101	
1000	10	1000	1111	0010	0101	
0100	11	0100	0011	1110	1001	
0001	01	0001	0110	1011	1100	



Ce code dans Σ^4 peut être représenté comme les sommets d'un hypercube.

- les codes ($\phi(c_i)$, à savoir les centres des classes) sont cerclés de noir.
- tous les sommets appartenant à la même classe sont de la même couleur.

Les capacités de correction de ce code sont bien nulles, car toute sphère de Hamming centré en un code c_i contient un code associée à une autre région R_j ($i \neq j$).

Théorème 63 (Borne de Hamming)**.**
Soit C un $[n, p, d]$-code linéaire et $t = \lfloor (d-1)/2 \rfloor$ sa capacité de correction. Alors $\sum_{i=0}^{t} \binom{n}{i} \leq 2^{n-k}$

Démonstration:

Les mots de code de C étant de taille n,

- $\binom{n}{0} = 1$ = nombre de vecteurs ϵ_i tels que $w_H(\epsilon_i) = 0$.
- $\binom{n}{1}$ = nombre de vecteurs ϵ_i tels que $w_H(\epsilon_i) = 1$ (=nombre de façon de placer un 1 dans n bits).
- ...
- $\binom{n}{t}$ = nombre de vecteurs ϵ_i tels que $w_H(\epsilon_i) = t$ (=nombre de façon de placer t 1 dans n bits).

Or, le nombre de syndromes est 2^{n-p}.

En conséquence, si C a une capacité de correction de t, ce code doit nécessairement vérifier $\sum_{i=0}^{t} \binom{n}{i} \leq 2^{n-k}$. □

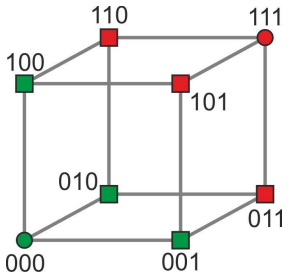
Autrement dit, le nombre de syndromes doit être plus grand ou égal au nombre d'erreurs corrigeables.

Définition 122 (Code parfait)**.**
Un code est dit parfait si la borne de Hamming est atteinte, à savoir si $\sum_{i=0}^{t} \binom{n}{i} = 2^{n-p}$.

Interprétation : les sphères de Hamming couvrent parfaitement l'espace Σ^n.

Exemple: code parfait.
Pour un code binaire à répétition $[3, 1, 3]$, $t = \lfloor (3-1)/2 \rfloor = 1$ et $\sum_{i=0}^{1} \binom{n}{i} = \frac{3!}{3!} + \frac{3!}{2!} = 1 + 3 = 4 = 2^{3-1}$.
$G = [\,1\ 1\ 1\,]$ et $H = \left[\begin{smallmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{smallmatrix}\right]$.

		0	1	c_i
ϵ_i	$\epsilon_i.H^t$	000	111	$\phi(c_i)$
000	00	000	111	
100	10	100	011	
010	11	010	101	
001	01	001	110	



Les boules de Hamming centrées aux mots du code recouvrent bien exactement tous les mots de la colonne associée à chaque mot.

Implémentation : On remarquera que :

- l'addition binaire bit à bit sans retenue peut être réalisée avec un OU exclusif : `0011 ^ 0101 = 0110`.
- la multiplication binaire bit à bit peut être réalisée avec un ET logique : `0011 & 0101 = 0001`.
- le calcul du poids `w` d'un mot binaire `x` s'effectue simplement comme :

```
w = 0;
while(x != 0) {
```

```
        if (x & 0x01) w++;
        x >>= 1;
    }
```

- le calcul de la distance de Hamming entre deux mots `u` et `v` se calcule comme le poids de `u ^ v`.

5 Codes de Hamming

5.1 Définition

Définition 123 (code de Hamming)**.**
Un code de Hamming est un $[2^m-1, 2^m-1-m, 3]$-code linéaire (à savoir un code linéaire pour lequel la capacité de détection est 3 et tel que la borne de Hamming est atteinte) et pour lequel la matrice de contrôle se construit comme toutes les combinaisons possibles de m bits non nuls.

Soit un (n, p)-code linéaire. Si la capacité de détection est 3, alors la capacité de correction est $t = \lfloor (3-1)/2 \rfloor = 1$. Si la borne de Hamming est atteinte avec $t = 1$, alors $\sum_{i=0}^{1} \binom{n}{i} = 1 + n = 2^{n-p}$ et en conséquence $n = 2^{n-p} - 1$. Notons maintenant $m = n - p$, donc $n = 2^m - 1$ et $p = n - m = 2^m - 1 - m$.

Remarquons maintenant que la matrice de contrôle H est de taille $(p + m) \times m$ (construite par bloc avec la transposée de la matrice de parité de taille $m \times p$ et de la matrice identité I_m). Or, $p + m = 2^m - 1 - m + m = 2^m - 1$. Mais les colonnes de cette matrice doivent être différente de 0 et distinctes deux à deux. Par ailleurs, ce sont des colonnes de taille m. Comme il n'y a que $2^m - 1$ façons différentes de construire des colonnes de taille m différentes et non nulles, ceci implique que toutes ces combinaisons sont dans H.

Exemple: code de Hamming $[7, 4, 3]$
Pour $m = 3$, $2^m - 1 = 7$ et $2^m - 1 - m = 4$.
Les colonnes de la matrice de contrôle sont $\begin{bmatrix} 0&0&0&1&1&1&1 \\ 0&1&1&0&0&1&1 \\ 1&0&1&0&1&0&1 \end{bmatrix}$.
On réarrange les colonnes afin d'avoir la matrice H sous forme systématique :

$$H_{\text{sys}} = \begin{bmatrix} 0&1&1&1&1&0&0 \\ 1&0&1&1&0&1&0 \\ 1&1&0&1&0&0&1 \end{bmatrix}.$$

On en tire la matrice de parité $P = \begin{bmatrix} 0&1&1&1 \\ 1&0&1&1 \\ 1&1&0&1 \end{bmatrix}$.
On en déduit la matrice génératrice sous forme systématique :

$$G_{\text{sys}} = \begin{bmatrix} 1&0&0&0&0&1&1 \\ 0&1&0&0&1&0&1 \\ 0&0&1&0&1&1&0 \\ 0&0&0&1&1&1&1 \end{bmatrix}.$$

5.2 Décodage

Proposition 64 (décodage simplifié d'un code de Hamming)**.**
Si les colonnes de la matrice de contrôle H sont triés par ordre croissant (où LSB = dernière ligne), alors le syndrome $s = y.H^t$ représente (en binaire) le numéro de la colonne (numérotée de 1 *à n) qu'il faut corriger sur y.*

Exemple: correction avec un code de Hamming $[7, 4, 3]$

La matrice de contrôle triée par ordre croissant des colonnes est $H = \begin{bmatrix} 0&0&0&1&1&1&1 \\ 0&1&1&0&0&1&1 \\ 1&0&1&0&1&0&1 \end{bmatrix}$.

Prenons un mot de code $x = 0100$, son code est $e = 0100101$.

- Ajoutons une erreur $\epsilon = 0001000$, $r = e + \epsilon = 0101101$.
  Calculons le syndrome : $s = (100)_2 = 4$. Le bit à corriger sur r est le $4^{\text{ème}}$ bit. $r' = 0100101 = e$. La correction est correcte ($r'.H^t = 000$).
- Ajoutons une erreur $\epsilon = 0000001$, $r = e + \epsilon = 0100100$.
  Calculons le syndrome : $s = (111)_2 = 7$. Le bit à corriger sur r est le $7^{\text{ème}}$ bit. $r' = 0100101 = e$. La correction est correcte ($r'.H^t = 000$).
- Ajoutons deux erreurs $\epsilon = 0100100$, $r = e + \epsilon = 0000001$.
  Calculons le syndrome : $s = (111)_2 = 7$. Le bit à corriger sur r est le $7^{\text{ème}}$ bit. Soit $e = 0000000 \neq r$. S'il y a plus d'une erreur, le mot reçu est décodé de manière incorrecte.

6 Codes de Golay

Le code de Golay est intéressant car il possède des propriétés remarquables.

Il est basé sur la constatation que $\sum_{i=0}^{3} \binom{23}{i} = 2^1 1$ (donc la borne de Hamming $\sum_{i=0}^{t} \binom{n}{i} = 2^{n-k}$ est atteinte pour $n = 23$, $t = 3$ et $n - k = 11$). En conséquence, il existe un code binaire parfait tel que :

- $n = 23$ (code binaire de 23 bits),
- $p = n - k = 23 - 11 = 12$ (mots sources de 12 bits),
- sa capacité de correction est de $t = 3$, et une capacité de détection de $d = 3.2 + 1 = 7$.

Nous ne donnerons pas plus de détail sur ce code dans le cadre de ce cours, mais son utilisation est similaire à celle d'un code de Hamming.

Comme les codes de Hamming, le code de Golay est aussi un code cyclique dont le polynome générateur est $g(x) = x^{11} + x^{10} + x^6 + x^5 + x^4 + x^2 + 1$ (voir le chapitre sur les codes cycliques).

Conclusion

Ce premier chapitre sur les codes correcteurs avait pour but de donner le contexte et le cadre de théorique de tel code.
Les codes linéaires représentent une première classe de code permettant d'effectuer une première approche des codes détecteurs-correcteurs d'erreur.
Nous abordons maintenant d'autres familles de code, qui même s'ils ne sont pas construits comme un code linéaire ont un code linéaire équivalent.