Retour TP 6 – Codes correcteurs d'erreurs

▶ **Prérequis** : *Chapitre 5 du poly.*

Nous allons utiliser dans ce TP le modèle suivant :

- On considère des messages sous forme d'une suite de bits ;
- Les bits seront regroupés par paquets de k bits avec k une puissance de 2 ;
- Chaque bit a une probabilité non nulle d'être altéré (et donc inversé).

1 Bit de parité

Pour pouvoir corriger une éventuelle erreur dans un bloc de bits, on doit ajouter une information redondante. En voici un premier exemple.

1. (**Bit de parité**) Coder une fonction qui découpe un message en blocs de 8 bits et qui transforme chaque bloc de 8 bits en un bloc de 9 bits en ajoutant à la fin un bit de sorte que la somme des bits du nouveau bloc de 9 bits soit paire.

   ▶ **Correction.–** Voir le Jupyter Notebook.

2. Tester votre fonction sur 01101011 et 00110101. Que se passe-t-il si un bit est modifié ? Et si deux bits sont modifiés ? Peut-on corriger lorsque l'on détecte une erreur ?

   ▶ **Correction.–** Si un bit est modifié, on peut le détecter mais pas si deux bits sont modifiés en général ! En revanche, on ne peut pas corriger les erreurs détectées !

On aimerait que notre algorithme ne rajoute pas trop d'informations au message initial, soit capable de détecter et corriger un nombre raisonnable d'erreurs mais aussi que les algorithmes de codage et de décodage soient efficaces.

2 Paramètres d'un code

Un bloc de k bits sera indifféremment appelé bloc, mot ou vecteur dans la suite. L'ensemble des mots de k bits n'est rien d'autre que $M_k = \mathbf{F}_2^k$. On parlera indifféremment de bits ou de lettres. Un mot de k bits sera noté $b_1 b_2 \cdots b_k$ ou $\begin{pmatrix} b_1 & b_2 & \cdots & b_k \end{pmatrix}$. Un code correcteur **de paramètre** (k, n) est une application injective $\Phi : M_k \to M_n$ appelée **encodage**. Le paramètre k est la **dimension** du code et n sa **longueur**. L'image C de Φ est appelée **image** du code et les éléments de C sont les **mots du code**.

3. Pourquoi demander que la fonction d'encodage soit injective ? Quelle relation existe entre k et n ?

   ▶ **Correction.–** Sinon, il existerait deux mots m et m' tels que $\Phi(m) = \Phi(m') = c$. Après réception de c même en supposant que le canal de transmission ne fasse pas d'erreur, on ne saurait alors dire si le mot envoyé est m ou m' ! On a alors $k \leqslant n$.

4. Quelle est la dimension et la longueur de l'ajout du bit de parité ?

   ▶ **Correction.–** Dans le cas de mots de 8 bits, on obtient un code de paramètre $(8, 9)$.

5. (**Code de répétition pure**) Considérons l'application $\Phi : M_1 \to M_3$ définie par $\Phi(0) = 000$ et $\Phi(1) = 111$. Expliciter toutes les notions ci-dessus au cas de ce code.

   ▶ **Correction.–** On a un code de paramètres $(1, 3)$.

6. Préciser ce que peuvent devenir les mots 000 et 111 après 0,1 et 2 erreurs. Parmi les mots trouvés, quels sont ceux qui sont des mots du code de répétition pure ci-dessus ? Combien d'erreurs ce code peut-il détecter ? Corriger ?

   ▶ **Correction.–** Après une erreur le mot 000 peut devenir $100, 010$ ou 001 tandis que 111 peut devenir $011, 101$ ou 110 et après deux erreurs, 000 peut devenir $000, 110, 011$ ou 101 tandis que 111 peut devenir $111, 001, 100$ ou 010. On constate que l'on peut donc détecter deux erreurs mais pas les corriger (par exemple le mot 100 reçu peut provenir de 000 après une erreur ou de 111 après deux erreurs). On peut en revanche corriger une erreur si l'on sait qu'on a au plus une erreur.

On appelle **poids de Hamming** de m et on note $\omega(m)$ le nombre de lettres non nulles de m.

7. Coder une fonction distance de Hamming et une fonction poids de Hamming. Tester vos fonctions sur 01100111 et avec les couples $(0001001, 0101001)$ et $(0000110, 0001100)$.

   ▶ **Correction.–** Voir le Notebook !

Soit Φ une fonction d'encodage d'image C. On appelle **capacité de détection** de Φ et on note e_d le plus grand nombre d'erreurs que Φ permet de détecter à coup sûr. On appelle **capacité de correction** de Φ et on note e_c le plus grand nombre d'erreurs que Φ permet de corriger à coup sûr. On appelle **distance minimale** de Φ et on note d_Φ la plus petite distance de Hamming non nulle entre deux mots du code.

10. Rappeler les relations entre ces différentes quantités.

    ▶ **Correction.–** On a $e_d = d_\Phi - 1$ (cela assure que les boules de centre les mots du code et de rayon e_d ne contiennent chacune qu'un seul mot du code) et $e_c = \lceil \frac{d_\Phi - 1}{2} \rceil$ d'après le cours.

11. Déterminer ces paramètres dans le cas du bit de parité et de la répétition pure.

▶ **CORRECTION.–** On obtient $d_\Phi = 2$ pour le bit de parité (soit théoriquement soit via `Sage`) donc on retrouve bien $e_d = 1$ et $e_c = 0$. Pour le code de répétition pure, on a $d_\Phi = 3$ donc $e_d = 2$ et $e_c = 1$.

3 Codes linéaires

Un code Φ de paramètres (k, n) est dit **linéaire** si l'injection Φ est une application linéaire. Autrement dit s'il existe une matrice $G \in \mathcal{M}_{k,n}(\mathbf{F}_2)$ de rang k telle que

$$\forall m \in M_k, \quad \Phi(m) = mG.$$

La matrice G est appelée **matrice génératrice** du code Φ.

12. Les codes du bit de parité et de répétition pure sont-ils linéaires ? Si oui, préciser leur matrice génératrice.

▶ **CORRECTION.–** Oui le code de parité est linéaire de matrice

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

et le code de répétition pure est également linéaire de matrice

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

13. Étudier le code linéaire ayant

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

comme matrice génératrice. Préciser en particulier ses paramètres, C et donner la liste des distances de Hamming, d_Φ ainsi que e_d ete_c.

▶ **CORRECTION.–** On a un code de paramètres $(2, 4)$. Les mots du code sont

$$\begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 \end{pmatrix} G = \begin{pmatrix} 1 & 0 & 1 & 1 \end{pmatrix}, \quad \begin{pmatrix} 0 & 1 \end{pmatrix} G = \begin{pmatrix} 0 & 1 & 0 & 1 \end{pmatrix} \quad \text{et} \quad \begin{pmatrix} 1 & 1 \end{pmatrix} G = \begin{pmatrix} 1 & 1 & 1 & 0 \end{pmatrix}.$$

On en déduit que $d_\Phi = 2$ et donc $e_d = 1$ et $e_d = 0$.

14. (**Bornes de Singleton et de Hamming**) Soit Φ un code linéaire de paramètre (k, n) d'image C. Rappeler la borne de Singleton et démontrer la borne de Hamming suivante

$$\sum_{i=0}^{e_c} C_n^i \leqslant 2^{n-k}.$$

▶ **CORRECTION.–** On a $d_\Phi \leqslant n - k + 1$ pour la borne de Singleton. Pour la borne de Hamming, voir la Remarque 5.3 du polycopié de cours !

Un code pour lequel on a égalité dans la borne de Singleton est dit **MDS (Maximum Distance Separable)** tandis qu'un code pour lequel on a égalité dans le borne de Hamming est dit **parfait**.

15. Donner un minorant sur la longueur d'un code linéaire de dimension k détectant d erreurs.

▶ **CORRECTION.–** On sait que $n \geqslant k + d_\Phi - 1 = k + e_d$. Si l'on suppose que $e_d = d$, on a alors $n \geqslant k + d$.

16. Les codes bit de parité, répétition pure et de la question 13 sont-ils MDS ? Parfaits ?

▶ **CORRECTION.–** Pour le code de parité, on obtient $d_\Phi = 2 = n-k+1$ donc le code est MDS. Par ailleurs, on a $\sum_{i=0}^{0} C_9^i = 1 < 2^{9-8} = 2$ donc le code n'est pas parfait !

Pour le code de répétition pure, $d_\Phi = 3 = n - k + 1$ donc le code est MDS et $\sum_{i=0}^{1} C_3^i = 1 + 3 = 4 = 2^{3-1} = 4$ donc le code est parfait !

17. Implémenter une fonction qui prend en argument la matrice génératrice d'un code linéaire et renvoie les paramètres du code d_Φ, e_d et e_c et si le code est parfait ou MDS. Tester votre fonction sur la matrice

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

Comparer avec les commandes préexistantes de `Sage` suivantes

Argument(s)	Commmande	Sortie
$G \in \mathcal{M}_{k,n}(\mathbf{F}_q)$	`LinearCode(G)`	code de matrice génératrice G
code `C`	`C.generator_matrix()`	matrice génératrice de `C`
code `C`	`C.parity_check_matrix()`	matrice de contrôle de `C`
mot `c` de `C`	`c.hamming_weight()`	poids de Hamming de `c`
code `C`	`C.minimum_distance()`	distance minimale de `C`
$r, q \geqslant 1$	`codes.HammingCode(GF(q),r)`	code de Hamming de longueur $\frac{q^r-1}{q-1}$
$n \geqslant 1, P \in \mathbf{F}_q[X]$	`codes.CyclicCode(n,P)`	code cyclique $(P) \subseteq (\mathbf{F}_q)^n$

Tester la perfection des codes de répétition de longueur inférieure à 32. Faire une conjecture.

▶ **CORRECTION.–** Voir le Notebook! Le code de répétition de longueur n consiste à prendre un mot de 1 bit et à répéter ce bit n fois. Une matrice génératrice est alors une ligne de n uns. On conjecture que les codes de répétition parfaits sont ceux de longueur impaire (de paramètres $(1, 2e_c + 1)$).

Un code Φ de paramètres (k, n) est dit **systématique** si pour tout $m \in M_k$, le mot m est un préfixe de $\Phi(m)$.

18. Montrer qu'un code linéaire de paramètre (k, n) est systématique si, et seulement si, sa matrice génératrice est de la forme $\begin{pmatrix} I_k & G' \end{pmatrix}$ où $G' \in \mathcal{M}_{k,n-k}(\mathbf{F}_2)$.

▶ **CORRECTION.–** C'est immédiat!

19. Soit Φ un code systématique de paramètres (k, n) et de matrice génératrice $\begin{pmatrix} I_k & G' \end{pmatrix}$. Rappeler pourquoi la matrice $\begin{pmatrix} {}^tG' & I_{n-k} \end{pmatrix}$ est une matrice de contrôle de Φ.

▶ **CORRECTION.–** C'est dans le poly!

20. Montrer que la distance d_Φ pour un code linéaire systématique est le nombre minimal de colonnes linéairement dépendantes de H où H est une matrice de contrôle.

▶ **CORRECTION.–** C'est aussi dans le poly!

21. Implémenter une fonction qui prend en entrée un code linéaire systématique à partir de sa matrice génératrice et renvoie une matrice de contrôle. Donner les matrices de contrôles des codes bits de parité, répétition pure et de la question 13.
Écrire une une fonction qui prend en entrée un code linéaire systématique à partir de sa matrice de contrôle et renvoie une matrice génératrice.

▶ **CORRECTION.–** Voir le Notebook!

Soit Φ un code de paramètres (k, n) et de matrice génératrice G et de contrôle H. On se fixe un mot x de longueur k. Le mot de code correspondant est $y = \Phi(x)$. S'il y a eu des erreurs durant la transmission, on reçoit un mot z. On appelle **mot erreur** associé à z le mot e tel que $z = y + e$. On rappelle que le **syndrome** de z est le mot z^tH. On voit alors immédiatement que $z^tH = e^tH$ et que le syndrome ne dépend que de l'erreur. L'ensemble des syndromes de z, noté S_z est appelé **classe** de z. Le principe du décodage est alors le suivant :

- On reçoit le mot z transmis avec d'éventuelles erreurs.
- On calcule la syndrome $s = z^tH$ de z.
- Si $s = 0$, on ne détecte pas d'erreur[1] et on retourne $\Phi^{-1}(z)$.
- Sinon on recherche l'erreur e de plus petit poids possible[2] telle que $e^tH = s$.
- Si $\omega(e) \leqslant e_c$, on retourne $\Phi^{-1}(z + e)$.
- Sinon retourner "*Impossible de corriger les erreurs*".

Il nous reste à expliquer comment déterminer e.

22. Rappeler pourquoi chaque syndrome s est associé à au plus un mot $m \in \mathbf{F}_2^n$ de poids $\omega(m) \leqslant e_c$.

▶ **CORRECTION.–** Si un syndrome est associé à deux mots du code m, m' de poids $\leqslant e_c$, alors $m^tH = {m'}^tH$ et donc ${}^tm - {}^tm'$ est dans le noyau de H, donc $m - m'$ appartient au code C. On a alors

$$\omega(m - m') \leqslant \omega(m) + \omega(m') \leqslant 2e_c \leqslant d_\Phi - 1$$

ce qui implique $m = m'$.

1. Ce qui ne signifie pas pour autant qu'il n'y en a pas! Pensez au bit de parité!
2. Donc la plus probable!

23. Que se passe-t-il dans le cas d'une seule erreur ? Par exemple prendre le code de matrice génératrice

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

et décoder le mot 0101 en sachant qu'il y a eu au plus une erreur.

▶ **Correction.–** Dans ce cas, il existe i tel que $z = y + e_i$ avec e_i le vecteur nul sauf en i-ème position où on a un 1. On a alors que le syndrome est simplement donné par la i-ème colonne de H. On peut alors localiser facilement l'erreur. Dans l'exemple, on calcule via `Sage` la matrice de contrôle et le syndrome pour obtenir la première colonne de H. On en déduit qu'il y a eu une erreur sur le premier bit et on corrige donc le mot en 1101 (qui est bien un mot du code).

4 Décodage

Pour trouver e, on va construire une **table de décodage**. Soit Φ un code linéaire de paramètres (k, n). L'ensemble des syndromes possibles est donc $\mathbf{F}_2^{n-k}$. Pour établir cette table de décodage, on liste tous les syndromes de Φ puis tous les mots z de $\mathbf{F}_2^n$ par poids croissant. Pour chaque z, on calcule alors $z^t H$ et au syndrome s on associe le premier mot z obtenu tel que $z^t H = s$. On arrête alors ce procédé dès qu'on a associé un mot à chaque syndrome et le mot associé à un syndrome est alors l'erreur de poids minimal donnant s.

24. Implémenter une fonction table de décodage et la tester sur le code bit de parité, répétition pure et le code de la question 13. Décoder alors respectivement les mots 101010100, 101 et 1100.

▶ **Correction.–** Voir le Notebook !

5 Codes de Hamming

Un **code de Hamming** est un code parfait de paramètres $(2^r - r - 1, 2^r - 1)$ pour $r \geqslant 2$ dont une matrice de contrôle est obtenu par n'importe quelle énumération en colonne de tous les mots de r bits non nuls [3].

25. Étudier et implémenter le code de Hamming capable de corriger une erreur et de paramètres $r = 2$.

▶ **Correction.–** On peut voir en utilisant la question **20.** qu'un code de Hamming a toujours une distance minimale de 3. En effet, on a aucune colonne nulle donc $d_\Phi \geqslant 2$. Par ailleurs, on n'a aucune colonne identique donc $d_\Phi \geqslant 3$. Enfin, en faisant la somme de deux colonnes ne contenant qu'un seul 1, on obtient un mot avec deux un, donc une autre colonne et donc le rang est inférieur à 3 donc égal à 3 et $d_\Phi = 3$. On a donc toujours $e_c = 1$ et le code est toujours parfait car

$$\sum_{i=0}^{1} C_{2^r-1}^i = 2^r = 2^{2^r-1-(2^r-r-1)}.$$

Voir le Notebook.

26. Étudier et implémenter le code de Hamming de matrice génératrice

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

et décoder le mot 1011101.

▶ **Correction.–** Voir le Notebook !

Les codes parfaits sont rares et on peut démontrer que les seuls codes linéaires parfaits sont les codes de répétition de paramètre $(1, 2e_c+1)$, les codes de Hamming et le code de Golay G_{23}.

6 Codes cycliques

27. Rappeler pourquoi se donner un code cyclique C de longueur n revient à se donner un polynôme unitaire P divisant $X^n - 1$, que l'on appelle **polynôme générateur** de C, et tel que

$$C = \{\pi(PQ) \ : \ \deg(Q) \leqslant n - \deg(P) - 1\}$$

avec $\pi : \mathbf{F}_2[X] \to A$ la surjection canonique. Donner une matrice génératrice du code.

3. Le minitel codait notamment ses données avec un code de Hamming de paramètre $(120, 128)$, soit $r = 7$.

▶ **Correction.–** C'est dans le polycopié pour la première partie ! Notons alors (car P divise $X^n - 1 = X^n + 1$) $P = 1 + \sum_{i=1}^{r} a_i X^i$.

Une matrice génératrice est immédiatement donnée par

$$\begin{pmatrix} 1 & a_1 & \cdots & a_r & 0 & 0 & \cdots & 0 \\ 0 & 1 & a_1 & \cdots & a_r & 0 & \cdots & 0 \\ & & \ddots & & & \ddots & & \\ 0 & \cdots & \cdots & 1 & a_1 & \cdots & & a_r \end{pmatrix}.$$

28. Écrire un programme qui détermine si un code est cyclique. Quels sont les codes présentés plus haut qui sont cycliques ? Donner le cas échéant les polynômes générateurs. Déterminer les codes cycliques de longueur 7 et en donner tous les paramètres.

▶ **Correction.–** Voir le Notebook ! Un code cyclique de longueur 7 correspond à un diviseur de $X^7 - 1 = (X-1)(X^6 + X^5 + X^4 + X^3 + X^2 + X + 1)$. On a donc le code nul associé à $X^7 - 1$, le code identité associé à 1 et le code associé à $X - 1$ de paramètres $(6, 7)$ et le code associé à $X^3 + X + 1$ et à $X^3 + X^2 + 1$ qui sont de paramètres $(3, 7)$.

29. Tester `codes_CyclicCode` avec $1, X-1, \frac{X^n-1}{X-1}$. Que reconnaît-on ? Que se passe-t-il avec un polynôme qui ne divise pas $X^n - 1$?

▶ **Correction.–** Voir le Notebook !

30. Trouver tous les polynômes irréductibles de degré 3 sur $\mathbf{F}_2$ et vérifier que les codes cycliques de longueur 7 qu'ils induisent sont équivalents.

▶ **Correction.–** Voir le Notebook !

L'opération de codage est alors encore plus simple ! Elle consiste à utiliser la bijection

$$\begin{cases} V & \longrightarrow & C \\ Q & \longmapsto & \pi(PQ) \end{cases}$$

avec V l'espace vectoriel de dimension $n - \deg(P)$ des polynômes de $\mathbf{F}_2[X]$ de degré inférieur à $n - \deg(P) - 1$. L'opération de codage consiste donc en pratique à une multiplication polynomiale modulo $X^n - 1$.
Le décodage se fait de façon analogue en déterminant le syndrome. Le syndrome d'un mot est alors défini comme le reste de la division euclidienne de ce mot par le polynôme générateur.

31. Donner une matrice de contrôle.

▶ **Correction.–** Cela résulte du polycopié page 35. Le codage d'un mot m sous forme polynomiale $P_m(X)$ est donné par $P(X)P_m(X)$ modulo $X^n - 1$. Si on pose $P(X)Q(X) = X^n - 1$ avec $Q(X) = \sum_{i=0}^{n-r} b_i X^i$, alors on vérifie qu'une matrice de contrôle est donnée par

$$\begin{pmatrix} b_{n-r} & b_{n-r-1} & \cdots & 1 & 0 & 0 & \cdots & 0 \\ 0 & b_{n-r} & b_{n-r-1} & \cdots & 1 & 0 & \cdots & 0 \\ & & \ddots & & & \ddots & & \\ 0 & \cdots & \cdots & b_{n-r} & b_{n-r-1} & \cdots & & b_0 \end{pmatrix}.$$

On a donc l'algorithme suivant :

- On détermine le syndrome $s = s_0$ de z par division euclidienne et on pose $i = 0$.
- Si $\omega(s_i) \leqslant e_c$, alors l'erreur est donnée par $e_i = X^{n-i}(s_i, \mathbf{0})$ et on retourne $m = z - e_i$.
- Sinon, on pose $i = i + 1$, on calcule s_i le syndrome de $X^i e_i$ et on recommence jusqu'à ce que $i = n$.
- Si $i = n$, l'erreur ne peut pas être corrigée.

32. Justifier que l'algorithme ci-dessus fonctionne.

▶ **Correction.–** Voir http ://math.univ-lyon1.fr/ caldero/Melvyn.pdf.

33. Implémenter et étudier un code cyclique de paramètres $(7, 15)$ et de polynôme générateur $X^8 + X^7 + X^6 + X^4 + 1$ et décoder le mot correspondant à $X^{11} + X^{10} + X^7 + X^6 + X^5 + X^4 + X^3 + X + 1$.

▶ **Correction.–** Voir le Notebook !

Les codes cycliques permettent de corriger e_c erreurs et d'en détecter $\deg(P)$ sous réserve qu'elles n'apparaissent pas n'importe où. Ces codes sont notamment utilisés dans certains codes barres, disques durs ou lors de vols spatiaux.

7 Complexité

▶ **Correction.–** Voir ici.

8 Bibliographie

Si vous souhaitez approfondir tout cela, je vous renvoie aux références Beck, *Objectif Agrégation* et Demazure, *Cours d'algèbre*.