وزارة التربية الوطنية
والتعليم الأولي والرياضة

CRMEF OUJDA
ASSOCIATION SPORTIVE SCOLAIRE

Codes Correcteurs d'Erreur : Hamming (7,4) & Reed-Solomon

CODES CORRECTEURS D'ERREUR

Réalisé par :

QAZAT MOHAMED

Encadré par :

Pr. DERFOUFI YOUNES

Les membres du jury : - *Pr. DERFOUFI YOUNES*
-Pr. SALHI MOHAMED

Soutenu le : : / /

Année Universitaire : 2025/2026

Résumé

Ce projet présente une étude rigoureuse de deux codes correcteurs d'erreur fondamentaux : le **code de Hamming (7,4)** et le **code de Reed-Solomon**. Après avoir rappelé les bases théoriques communes (distance de Hamming, codes linéaires, algèbre dans les corps finis), nous décrivons en détail les mécanismes d'encodage et de décodage de chaque code, accompagnés d'exemples numériques complets. Nous établissons les propriétés mathématiques essentielles — distance minimale, capacité de correction, optimalité — et proposons une comparaison argumentée des deux approches. Des remarques sur l'implémentation algorithmique et les applications industrielles complètent l'analyse.

Mots-clés : *code correcteur, distance de Hamming, code linéaire, corps fini,* $\mathrm{GF}(2^8)$*, polynôme générateur, syndrome, algorithme d'Euclide étendu, formule de Forney.*

Table des matières

Introduction

Toute transmission ou tout stockage de données est confronté à un problème fondamental : le bruit. Qu'il s'agisse d'une rayure sur un disque optique, d'une perturbation électromagnétique sur un réseau de communication ou d'un défaut de cellule mémoire, des **erreurs** peuvent s'introduire dans le message et corrompre l'information. La **théorie des codes correcteurs** fournit les outils mathématiques pour détecter et corriger ces erreurs, en exploitant une redondance judicieusement construite.

L'histoire de la discipline débute en 1948 avec les travaux de **Claude Shannon** qui pose les fondements théoriques de la communication. En 1950, **Richard Hamming** propose le premier code pratique permettant de corriger une erreur binaire. En 1960, **Irving Reed** et **Gustave Solomon** généralisent ce principe en construisant des codes capables de corriger plusieurs erreurs sur des alphabets non-binaires.

Ces deux codes — Hamming et Reed-Solomon — restent aujourd'hui parmi les plus utilisés dans l'industrie. Le code de Reed-Solomon équipe les CD, les DVD, les codes QR, les communications satellitaires et les systèmes RAID. Le code de Hamming est présent dans les mémoires ECC et les protocoles de communication bas niveau.

Objectifs du projet

1. Rappeler les bases mathématiques : distance de Hamming, codes linéaires, corps finis.
2. Présenter le code de Hamming (7,4) de façon rigoureuse et illustrée.
3. Développer le code de Reed-Solomon : encodage, syndromes, algorithme d'Euclide, formule de Forney.
4. Comparer les deux approches et discuter leurs domaines d'application.

Chapitre 1

Théorie générale des codes correcteurs

1.1 Alphabet, message, encodeur

Un code correcteur s'articule autour de plusieurs concepts fondamentaux. Nous les définissons ci-dessous de manière formelle.

Définition 1.1 — Paramètres d'un code

- L'**alphabet** est un corps fini $\mathbb{F}_q$ à q éléments, muni d'opérations de somme et produit.
- Le **message** est un vecteur de k symboles : $A \in \mathbb{F}_q^k$.
- L'**encodeur** est une injection $\text{enc} : \mathbb{F}_q^k \to \mathbb{F}_q^n$ ajoutant $p = n - k$ symboles de contrôle.
- Le **code** $\mathcal{C} = \{\text{enc}(A) \mid A \in \mathbb{F}_q^k\}$ est l'image de l'encodeur.
- Le **mot reçu** $D = C + E$ est le mot du code C potentiellement affecté par un vecteur d'erreurs E.

On résume les paramètres d'un code par la notation $\mathcal{C}(n, k, d_{\min})_q$, où n est la longueur du mot de code, k la dimension du message, $d_{\min}$ la distance minimale et q le cardinal de l'alphabet.

1.2 Distance de Hamming

Définition 1.2 — Distance de Hamming

Soient $c_1, c_2 \in \mathbb{F}_q^n$. La **distance de Hamming** est définie par :

$$d(c_1, c_2) = \#\{i \in 1, n \mid c_{1,i} \neq c_{2,i}\}$$

C'est le **nombre de positions** où les deux mots diffèrent. Le **poids** d'un mot c est $w(c) = d(c, \mathbf{0})$.

Propriétés de la distance de Hamming

Proposition 1.1

La distance de Hamming est une **distance** au sens mathématique — elle vérifie :

- **Symétrie :** $d(c_1, c_2) = d(c_2, c_1)$
- **Séparation :** $d(c_1, c_2) = 0 \iff c_1 = c_2$
- **Inégalité triangulaire :** $d(c_1, c_3) \leq d(c_1, c_2) + d(c_2, c_3)$

Démonstration de l'inégalité triangulaire : Pour chaque position i, si $c_{1,i} \neq c_{3,i}$ alors $c_{1,i} \neq c_{2,i}$ ou $c_{2,i} \neq c_{3,i}$ (ou les deux). En sommant sur toutes les positions, on obtient $d(c_1, c_3) \leq d(c_1, c_2) + d(c_2, c_3)$. ■

Exemple illustratif : Considérons les mots binaires $c_1 = 1010101$ et $c_2 = 1100110$. Les positions qui diffèrent sont les positions 2, 3, 5 et 6 (en numérotant à partir de 1), donc $d(c_1, c_2) = 4$.

1.3 Codes t-correcteurs et codes parfaits

Définition 1.3 — Code t-correcteur

Un code $\mathcal{C}$ est dit **t-correcteur** si les boules de Hamming de rayon t centrées en chaque mot du code sont **disjointes** :

$$\forall (c_1, c_2) \in \mathcal{C}^2,\ c_1 \neq c_2 \implies B(c_1, t) \cap B(c_2, t) = \emptyset$$

où $B(c, t) = \{a \in \mathbb{F}_q^n \mid d(c, a) \leq t\}$. Un code t-correcteur **parfait** est un code pour lequel ces boules recouvrent tout $\mathbb{F}_q^n$.

Proposition 1.2 — Borne de Hamming

Pour tout code $\mathcal{C}$ t-correcteur sur $\mathbb{F}_q^n$, on a :

$$|\mathcal{C}| \cdot V_t \leq q^n \qquad \text{avec} \qquad V_t = \sum_{\ell=0}^{t} \binom{n}{\ell} (q-1)^\ell$$

Le code est parfait si et seulement si l'égalité est atteinte.

Proposition 1.3 : Un code de distance minimale $d_{\min}$ est exactement $\lfloor (d_{\min}-1)/2 \rfloor$-correcteur. En effet, si au plus $t = \lfloor (d_{\min}-1)/2 \rfloor$ erreurs se produisent, la boule de rayon t autour du mot reçu ne peut contenir qu'un seul mot du code — le mot d'origine.

1.4 Codes linéaires

Définition 1.4 — Code linéaire

Un code $\mathcal{C}$ est **linéaire** de longueur n et dimension k sur $\mathbb{F}_q$ si $\mathcal{C}$ est un sous-espace vectoriel de $\mathbb{F}_q^n$ de dimension k.
Propriété fondamentale : la somme de deux mots du code est un mot du code ; le mot nul appartient au code.

Matrice génératrice $G \in \mathcal{M}(n \times k, \mathbb{F}_q)$ **:** dont les colonnes forment une base de $\mathcal{C}$. L'encodage s'écrit $C = G \cdot A$.

Matrice de contrôle $H \in \mathcal{M}((n-k) \times n, \mathbb{F}_q)$ **:** telle que pour tout mot de code c, $H \cdot c = \mathbf{0}$. Elle permet de **vérifier** si un mot reçu appartient au code.

Proposition 1.4

Pour un code linéaire, la distance minimale vaut :

$$d_{\min} = \min\{ w(c) \mid c \in \mathcal{C} \setminus \{\mathbf{0}\} \}$$

C'est le **poids minimal** d'un mot non nul du code.

Chapitre 2

Structure des corps finis

Les codes de Reed-Solomon travaillent dans un **corps fini** (aussi appelé **corps de Galois**). Nous rappelons ici les constructions et propriétés essentielles.

2.1 Corps premiers $\mathbb{F}_p$

Soit p un entier naturel. On construit l'anneau $\mathbb{Z}/p\mathbb{Z} = \{0, 1, \ldots, p-1\}$ muni de l'addition et de la multiplication modulo p.

Théorème 2.1

L'anneau $(\mathbb{Z}/p\mathbb{Z}, +, \times)$ est un **corps** si et seulement si p est un **nombre premier**. Dans ce cas, on le note $\mathbb{F}_p$.

Preuve (sens direct) : si p est premier et $a \not\equiv 0\ [p]$, alors $\gcd(a, p) = 1$. Par le théorème de Bézout, il existe $u, v \in \mathbb{Z}$ tels que $ua + vp = 1$, donc $ua \equiv 1\ [p]$, soit u est l'inverse de a. ■

2.2 Corps de Galois $\mathrm{GF}(p^n) = \mathbb{F}_p[X]/(P)$

Pour construire un corps à p^n éléments, on prend un **polynôme irréductible** P de degré n sur $\mathbb{F}_p$, et on construit l'anneau quotient $\mathbb{F}_p[X]/(P)$.

Théorème 2.2 — Corps de Galois

L'anneau $\mathbb{F}_p[X]/(P)$ est un corps si et seulement si P est **irréductible** dans $\mathbb{F}_p[X]$. Ce corps à p^n éléments est noté $\mathrm{GF}(p^n)$ ou $\mathbb{F}_{p^n}$.

Construction de $\mathrm{GF}(2^8)$ — l'exemple clé pour Reed-Solomon

Pour les applications pratiques (un octet = 8 bits), on travaille dans $\mathrm{GF}(2^8)$, corps à 256 éléments. On prend le polynôme irréductible standard :

$$P(X) = X^8 + X^4 + X^3 + X^2 + 1 \qquad (\text{sur } \mathbb{F}_2)$$

Un élément de $\mathrm{GF}(2^8)$ est représenté par un polynôme de degré ≤ 7 à coefficients dans $\{0, 1\}$, c'est-à-dire par un entier de 0 à 255. Par exemple : `0001001010` $= X^6 + X^3 + X^1$ représente l'entier 74.

Opération	**Dans** $\mathrm{GF}(2^8)$	**Exemple**
Addition	XOR bit à bit ($\oplus$)	$29 \oplus 45 = 48$
Soustraction	Identique à l'addition (car $-1 = 1$ dans $\mathbb{F}_2$)	$a - b = a \oplus b$
Multiplication	Produit polynomial mod $P(X)$	$3 \times 5 = 15$ dans $\mathrm{GF}(2^8)$
Division	Multiplication par l'inverse	$a/b = a \times b^{-1}$
Inverse	$a^{-1} = \alpha^{255-\log_\alpha(a)}$	Algorithme d'Euclide

2.3 Racine primitive et tables logarithmiques

Théorème 2.3 — Groupe multiplicatif cyclique

Le groupe multiplicatif $(\mathrm{GF}(2^8)^*, \times)$ est **cyclique**. Il existe un élément α (appelé **racine primitive** ou **générateur**) tel que :

$$\mathrm{GF}(2^8)^* = \{\alpha^0, \alpha^1, \alpha^2, \ldots, \alpha^{254}\}$$

Pour $\mathrm{GF}(2^8)$ avec $P(X) = X^8 + X^4 + X^3 + X^2 + 1$, on prend $\alpha = 2$.

Chapitre 3

Code de Hamming (7,4)

3.1 Construction et paramètres

Le code de Hamming est un code linéaire binaire (alphabet $\mathbb{F}_2 = \{0, 1\}$) défini par deux paramètres ($r \geq 2$) :

$$n = 2^r - 1, \qquad k = 2^r - r - 1, \qquad p = r \quad \text{(bits de contrôle)}$$

Pour $r = 3$, on obtient le code **Hamming(7,4)** : 4 bits de message, 3 bits de contrôle, 7 bits au total.

Paramètres du Hamming(7,4)

- $n = 7$: longueur du mot de code
- $k = 4$: bits d'information (message)
- $p = 3$: bits de parité (contrôle)
- $d_{\min} = 3$: distance minimale
- $t = 1$: nombre d'erreurs corrigibles
- **Code parfait** : les boules de rayon 1 couvrent tout $\mathbb{F}_2^7$

3.2 Encodage — Matrice génératrice

Les 7 positions du mot de code sont numérotées de 1 à 7. Les positions qui sont des **puissances de 2** (1, 2, 4) sont réservées aux bits de parité. Les autres positions (3, 5, 6, 7) reçoivent les bits de message.

Position	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Bit	p_1	p_2	d_1	p_3	d_2	d_3	d_4
Type	Parité	Parité	Donnée	Parité	Donnée	Donnée	Donnée

Les bits de parité sont définis par les règles de contrôle suivantes (en notation modulo 2) :

— p_1 contrôle les positions 1, 3, 5, 7 $\Rightarrow p_1 = d_1 \oplus d_2 \oplus d_4$

— p_2 contrôle les positions 2, 3, 6, 7 $\Rightarrow p_2 = d_1 \oplus d_3 \oplus d_4$

— p_3 contrôle les positions 4, 5, 6, 7 $\Rightarrow p_3 = d_2 \oplus d_3 \oplus d_4$

La **matrice génératrice** G est la matrice 7×4 telle que le mot de code est $B = G \cdot A$:

Matrice génératrice G du Hamming(7,4)

$$G = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{matrix} \leftarrow p_1 \\ \leftarrow p_2 \\ \leftarrow d_1 \\ \leftarrow p_3 \\ \leftarrow d_2 \\ \leftarrow d_3 \\ \leftarrow d_4 \end{matrix}$$

Exemple numérique : Encodons le message $A = (1, 0, 1, 1)$.

$$p_1 = 1 \oplus 0 \oplus 1 = 0$$
$$p_2 = 1 \oplus 1 \oplus 1 = 1$$
$$p_3 = 0 \oplus 1 \oplus 1 = 0$$

Mot de code : $B = (0, 1, 1, 0, 0, 1, 1)$ (en ordre : $p_1, p_2, d_1, p_3, d_2, d_3, d_4$).

3.3 Détection et correction d'erreurs

La matrice de contrôle H

La **matrice de contrôle** H est une matrice 3×7 dont les colonnes sont les représentations binaires des entiers 1 à 7 :

Matrice de contrôle H du Hamming(7,4)

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

On vérifie : ${}^tH \cdot G = \mathbf{0}$ (propriété fondamentale des codes linéaires).

Calcul du syndrome

Soit D le mot reçu. On calcule le **syndrome** :

$$S = H \cdot D \quad \text{(calcul dans } \mathbb{F}_2\text{)}$$

Deux cas se présentent :

— $S = \mathbf{0}$: aucune erreur détectée, D est un mot du code valide.

— $S \neq \mathbf{0}$: il y a une erreur. S est l'écriture binaire de la **position de l'erreur** (Si ton calcul te donne le syndrome 110 (qui vaut 6 en décimal), cela signifie que c'est exactement le 6ème bit du mot reçu qui a été altéré) . Il suffit de retourner le bit en cette position.

Propriété remarquable de H

Les colonnes de H sont exactement les représentations binaires de 1 à 7. Ainsi $H \cdot E_i =$ représentation binaire de i. Si une seule erreur s'est produite en position i, le syndrome identifie directement cette position.

Exemple de correction : Supposons que le mot $B = (0, 1, 1, 0, 0, 1, 1)$ a subi une erreur en position 3 : $D = (0, 1, 0, 0, 0, 1, 1)$.

Étape	**Résultat**
Calcul $H \cdot D$	$S = H \cdot D = (1, 1, 0)^T$
Lecture du syndrome	$(0, 1, 1)_2 = 3$
Position de l'erreur	Position 3
Correction	Retourner le bit 3 : $(0, 1, 1, 0, 0, 1, 1)$ ✓

3.4 Propriétés fondamentales

Théorème 3.1 — Propriétés du code de Hamming (7,4)

1. **Code linéaire :** $\mathcal{C}$ est un sous-espace vectoriel de $\mathbb{F}_2^7$ de dimension 4.
2. **Distance minimale** $d_{\min} = 3$ **:** tout mot non nul a au moins 3 bits à 1.
3. **Code 1-correcteur :** peut corriger une erreur binaire.
4. **Code parfait :** $V_1 \times |\mathcal{C}| = (1 + 7) \times 16 = 128 = 2^7$. Les boules de rayon 1 couvrent $\mathbb{F}_2^7$ exactement.

Démonstration que $d_{\min} = 3$ **:** Soit $c \in \mathcal{C} \setminus \{\mathbf{0}\}$. Écrivons $c = G \cdot A$ avec $A \neq \mathbf{0}$.

— Si A a 1 bit non nul $\rightarrow$ c a au moins 3 bits non nuls.

— Si A a 2 bits non nuls $\rightarrow$ $w(c) \geq 3$.

— Si A a 3 ou 4 bits non nuls $\rightarrow w(c) \geq 3$ directement.

Par ailleurs, $G \cdot (1,0,0,0)^T = (1,1,1,0,0,0,0)$ qui a exactement poids 3. Donc $d_{\min} = 3$.
■

Code parfait : la borne de Hamming pour $t = 1$ vaut $V_1 = 1 + 7 = 8$. Le code a $2^4 = 16$ mots, et $16 \times 8 = 128 = 2^7$. La borne est atteinte.

Chapitre 4

Code de Reed-Solomon

Le code de Hamming que nous venons de voir est génial, mais il a un défaut majeur : **il ne corrige qu'une seule erreur**. Si une poussière ou une grosse rayure sur un CD détruit 5 bits d'affilée (s'appelle une "bouffée d'erreurs"), Hamming est totalement perdu. Les codes de Reed-Solomon (RS) ont été inventés pour corriger des erreurs multiples et groupées(Si une rayure détruit 8 bits consécutifs, l'algorithme de Reed-Solomon ne voit pas "8 erreurs". Il voit "1 seul symbole faux"). C'est l'algorithme qui a sauvé les CD, les DVD, l'ADSL et qui est aujourd'hui utilisé dans tous les QR codes . Ici, on change de dimension : on quitte les simples matrices binaires pour utiliser des polynômes et les fameux Corps de Galois.

4.1 Paramètres et structure

Le code de Reed-Solomon est un code linéaire sur un corps fini $\mathrm{GF}(q^m)$. Il est défini par deux paramètres n et k vérifiant :

$$n = q^m - 1, \qquad p = n - k, \qquad (p \text{ pair}, \ p = 2t)$$

Paramètres RS(n, k) sur $\mathrm{GF}(2^8)$

- n : longueur du mot de code (au plus 255 pour $\mathrm{GF}(2^8)$)
- k : nombre de symboles d'information
- $p = n - k$: symboles de parité (redondance)
- $t = p/2$: nombre d'erreurs corrigibles
- $d_{\min} = p + 1 = n - k + 1$: distance minimale (borne de Singleton atteinte !)

$\Rightarrow$ *code MDS*

Exemple pratique — RS(255, 223) : utilisé dans les systèmes de communication spatiale. $n = 255$, $k = 223$, $p = 32$ symboles de parité, corrige jusqu'à 16 erreurs d'octets.

4.2 Polynôme générateur

Soit α un **générateur** de $\mathrm{GF}(2^8)^*$ (racine primitive). Le **polynôme générateur** est :

$$G(X) = \prod_{i=1}^{p}(X - \alpha^i) = (X - \alpha)(X - \alpha^2)\cdots(X - \alpha^p)$$

Ce polynôme possède les propriétés suivantes :

— Il est de degré p.

— Ses racines sont $\alpha, \alpha^2, \ldots, \alpha^p$ — des éléments **connus** de $\mathrm{GF}(2^8)$.

— **Tout mot de code** $C(X)$ **est divisible par** $G(X)$, invariant central du décodage.

4.3 Encodage

Le message $A(X) = a_0 + a_1X + \cdots + a_{k-1}X^{k-1}$ est vu comme un polynôme de degré au plus $k-1$ sur $\mathrm{GF}(2^8)$. L'encodage produit un multiple de $G(X)$:

Algorithme d'encodage

1. **Décaler :** $T(X) = A(X) \cdot X^p$ (réserver p places pour la redondance)
2. **Diviser :** $T(X) = Q(X) \cdot G(X) + R(X)$ avec $\deg(R) < p$
3. **Mot de code :** $C(X) = T(X) - R(X) = Q(X) \cdot G(X)$

Garantie : $C(X) \bmod (G(X)) = 0$.

4.4 Détection d'erreurs — Calcul des syndromes

Soit $D(X) = C(X) + E(X)$ le polynôme reçu, où $E(X)$ est le polynôme d'erreur. On calcule les p **syndromes** en évaluant D aux racines du polynôme générateur :

$$S_j = D(\alpha^j) \qquad \text{pour } j = 1, 2, \ldots, p$$

Puisque $C(\alpha^j) = 0$ pour tout j, on a $S_j = E(\alpha^j)$.

Représentation formelle de l'erreur

L'erreur s'écrit $E(X) = \sum_{k=1}^{v} e_{i_k} X^{i_k}$ avec $v \leq t$ erreurs aux positions $i_1, \ldots, i_v$ et valeurs $e_{i_1}, \ldots, e_{i_v}$ dans $\mathrm{GF}(2^8)$. On pose $X_k = \alpha^{i_k}$ (localisateurs d'erreur). Le j-ième syndrome vaut :

$$S_j = \sum_{k=1}^{v} e_{i_k} \cdot (\alpha^{i_k})^j = \sum_{k=1}^{v} e_{i_k} \cdot X_k^j$$

4.5 Correction — Algorithme d'Euclide étendu

Les polynômes clés

Polynôme	Définition	Rôle
Syndrome $S(X)$	$\sum_{j=1}^{p} S_j X^{j-1}$	Données connues du décodeur
Localisateur $\Lambda(X)$	$\prod_k (1 - \alpha^{i_k} X)$	Racines $= \alpha^{-i_k}$ = positions
Évaluateur $\Omega(X)$	$\sum_k e_{i_k} \alpha^{i_k} \prod_{h \neq k} (1 - \alpha^{i_h} X)$	Valeurs d'erreur

Théorème 4.1 — Équation-clé

Les polynômes Λ et Ω vérifient l'équation-clé :

$$S(X) \cdot \Lambda(X) \equiv \Omega(X) \pmod{X^p}$$

avec $\deg(\Omega) < t$, $\deg(\Lambda) \leq t$ et $(\Lambda, \Omega) = 1$.
Ce théorème réduit le problème de décodage à la résolution d'une équation polynomiale

Résolution par l'algorithme d'Euclide étendu

Algorithme d'Euclide étendu (pour Λ et Ω)

$P_0 \leftarrow X^p, \quad P_1 \leftarrow S(X)$
$V_0 \leftarrow 0, \quad V_1 \leftarrow 1$
while $\deg(P_1) \geq t$ **do**
    $(Q, R) \leftarrow \text{DivisionEuclidienne}(P_0, P_1)$
    $(P_0, P_1) \leftarrow (P_1, R)$
    $(V_0, V_1) \leftarrow (V_1, V_0 - Q \cdot V_1)$
end while
$c \leftarrow 1/V_1(0)$
$\Lambda(X) \leftarrow c \cdot V_1(X)$
$\Omega(X) \leftarrow c \cdot P_1(X)$

Complexité : $O(t)$ divisions de polynômes de degré $\leq 2t \rightarrow O(t^2)$ opérations dans $\text{GF}(2^8)$.

4.6 Détermination des positions d'erreur

Les positions d'erreur s'obtiennent par la **recherche de Chien** : on évalue Λ en chaque élément non nul de $\mathrm{GF}(2^8)$:

$$\Lambda(\alpha^{-i}) = 0 \quad \Longleftrightarrow \quad \text{position } i \text{ est une position d'erreur}$$

On effectue donc 255 évaluations de Λ.

4.7 Calcul des valeurs d'erreur — Formule de Forney

Théorème 4.2 — Formule de Forney

Connaissant les positions d'erreur i_k, la valeur de l'erreur en cette position est :

$$e_{i_k} = -\frac{\Omega(\alpha^{-i_k})}{\Lambda'(\alpha^{-i_k})}$$

où $\Lambda'(X)$ est la dérivée formelle (c'est la notion de derivé dans un corps fini) de $\Lambda(X)$ dans $\mathrm{GF}(2^8)[X]$.

Preuve : En évaluant Ω et Λ' en α^{-i_k}, les produits correspondant aux autres positions ($h \neq k$) s'annulent, ne laissant que le terme en e_{i_k}.

On construit ensuite le polynôme d'erreur $E(X) = \sum_k e_{i_k} X^{i_k}$, et on corrige : $C(X) = D(X) - E(X)$ $(= D(X) + E(X))$. Dans $\mathrm{GF}(2^8)$, la soustraction est identique à l'addition (XOR).

4.8 Synthèse — Algorithme complet de décodage

Étape	Action	Formule
1	Calcul des syndromes	$S_j = D(\alpha^j)$, $j = 1$ à p
2	Vérification	Si tous $S_j = 0$: pas d'erreur
3	Équation-clé (Euclide)	Calculer $\Lambda(X)$ et $\Omega(X)$ tels que $S\Lambda \equiv \Omega \pmod{X^p}$
4	Recherche des racines de Λ	Chien search : trouver les α^{-i_k}
5	Formule de Forney	$e_{i_k} = -\Omega(\alpha^{-i_k})/\Lambda'(\alpha^{-i_k})$
6	Correction	$C(X) = D(X) \oplus E(X)$
7	Décodage	Extraire les coefficients d'ordre p à $n-1$

Théorème 4.3 — Propriétés du code Reed-Solomon

1. **Code linéaire** (n, k) **:** $\mathcal{C}$ est un sous-espace vectoriel de dimension k de $\mathrm{GF}(q^m)[X]$.
2. **Distance minimale** $d_{\min} = p + 1 = n - k + 1$ **:** le code atteint la borne de Singleton $\rightarrow$ code MDS (Maximum Distance Separable).
3. t**-correcteur avec** $t = \lfloor p/2 \rfloor$ **:** peut corriger jusqu'à t erreurs de symboles.
4. **Robustesse aux bouffées d'erreurs :** un symbole de $\mathrm{GF}(2^8) = 1$ octet entier corrompu compte comme 1 seule erreur.

Chapitre 5

Comparaison et applications

5.1 Tableau comparatif

Critère	**Hamming (7,4)**	**Reed-Solomon**
Alphabet	$\mathbb{F}_2 = \{0, 1\}$ (binaire)	GF(q^m), ex. GF(2^8)
Unité d'information	Bit	Symbole (souvent 1 octet)
Paramètres généraux	$n = 2^r - 1$, $k = 2^r - r - 1$	$n = q^m - 1$, k quelconque
Distance minimale	$d_{\min} = 3$	$d_{\min} = n - k + 1$
Erreurs corrigibles	$t = 1$ erreur binaire	$t = \lfloor (n-k)/2 \rfloor$ symboles
Erreurs détectables	2 erreurs	$n - k$ erreurs
Code parfait ?	Oui (pour $n = 2^r - 1$)	Non en général
Code MDS ?	Non	Oui (borne Singleton)
Décodage	Syndrome $H \cdot D$ simple	Euclide étendu + Forney
Complexité décodage	$O(n)$ — très rapide	$O(t^2)$ — polynomial
Bouffées d'erreurs	Vulnérable	Excellente résistance
Implémentation	Très simple	Modérément complexe

5.2 Limites théoriques

Deux bornes théoriques encadrent la qualité de tout code (n, k, d) sur $\mathbb{F}_q$:

— **Borne de Hamming (Sphere-Packing) :** un code t-correcteur vérifie $|\mathcal{C}| \cdot V_t \leq q^n$. Le code de Hamming atteint cette borne — il est parfait.

— **Borne de Singleton :** $d_{\min} \leq n - k + 1$ pour tout code linéaire. Reed-Solomon atteint cette borne — il est MDS (Maximum Distance Separable).

Remarque

Reed-Solomon est **optimal au sens de Singleton** : pour p symboles de redondance, il corrige $\lfloor p/2 \rfloor$ erreurs de symboles, quelle que soit la position des erreurs. Aucun code linéaire (n, k) ne peut avoir une distance minimale supérieure à $n-k+1$.

5.3 Applications industrielles

Code de Hamming

— **Mémoires ECC :** les modules RAM ECC utilisent le code de Hamming pour détecter et corriger les erreurs de bit dues aux radiations cosmiques ou aux défauts de fabrication.

— **Protocoles de communication :** utilisé en couche basse dans certains protocoles réseau pour une correction rapide d'un bit.

— **Cache processeur :** protection des caches L1/L2 dans les processeurs serveurs (Intel Xeon, AMD EPYC).

Code de Reed-Solomon

— **CD/DVD/Blu-Ray :** RS(28,24) en couche interne et RS(32,28) en couche externe (code croisé CIRC). Peut corriger jusqu'à 4000 octets consécutifs d'erreurs sur un CD.

— **Codes QR :** 4 niveaux de correction (7%, 15%, 25%, 30%). Le niveau H (30%) permet de lire un code QR même avec 30% de surface endommagée.

— **Communications spatiales :** RS(255,223) utilisé par la NASA pour les sondes Voyager, Cassini, et le réseau Deep Space Network.

— **Stockage RAID et codes d'effacement :** généralisations de Reed-Solomon pour les systèmes de stockage distribués (Ceph, Hadoop).

— **Wi-Fi et LTE :** codes apparentés (LDPC, Turbo codes) dérivant des mêmes principes algébriques.

Partie pratique

Afin d'illustrer concrètement les notions théoriques développées dans les chapitres précédents, une application web interactive nommée **CodecLab** a été réalisée. Cet outil permet de manipuler en temps réel les deux codes correcteurs étudiés — Hamming (7,4) et Reed-Solomon — et de visualiser leur comportement face à des erreurs de transmission.

Présentation générale

CodecLab se présente comme un atelier numérique dédié à la théorie des codes correcteurs d'erreurs. L'interface, sobre et épurée, s'organise autour de deux modules principaux accessibles depuis un même menu : un module *Codage / Décodage* et un module *QR Code*. L'utilisateur peut à tout moment choisir le code correcteur employé : Hamming (7,4) ou Reed-Solomon sur $GF(2^8)$, grâce à des indicateurs toujours visibles en haut de la page.

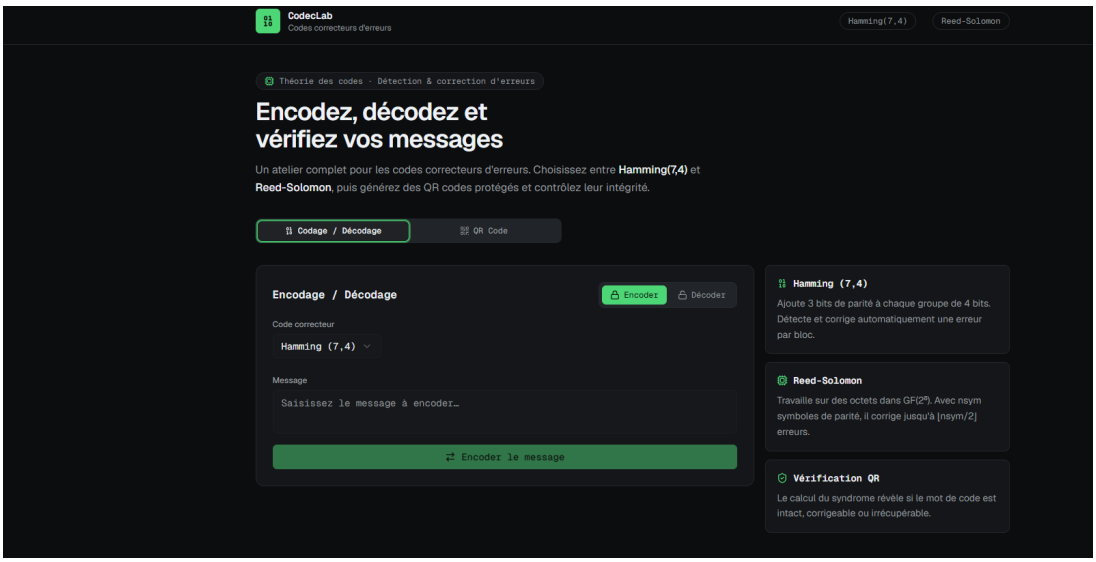

FIGURE 5.1 – Page d'accueil de l'application CodecLab : module Encodage avec le code de Hamming (7,4).

La page d'accueil rappelle les principes fondamentaux des deux codes disponibles : le code de Hamming (7,4), qui ajoute 3 bits de parité à chaque groupe de 4 bits et corrige automatiquement une erreur par bloc, ainsi que le code de Reed-Solomon, qui travaille sur

des octets dans $GF(2^8)$ et corrige jusqu'à $\lfloor n_{symbole}/2 \rfloor$ erreurs selon le nombre de symboles de parité choisi.

Module Codage / Décodage

Ce module permet d'encoder puis de décoder un message texte selon le code correcteur sélectionné. L'utilisateur choisit d'abord l'algorithme dans un menu déroulant : *Hamming (7,4)* ou *Reed-Solomon (GF 256)*. Dans ce second cas, un paramètre supplémentaire apparaît, le nombre de symboles de parité n_{sym}, dont la valeur détermine directement la capacité de correction du code (par exemple, $n_{sym} = 4$ permet de corriger 2 erreurs de symboles).

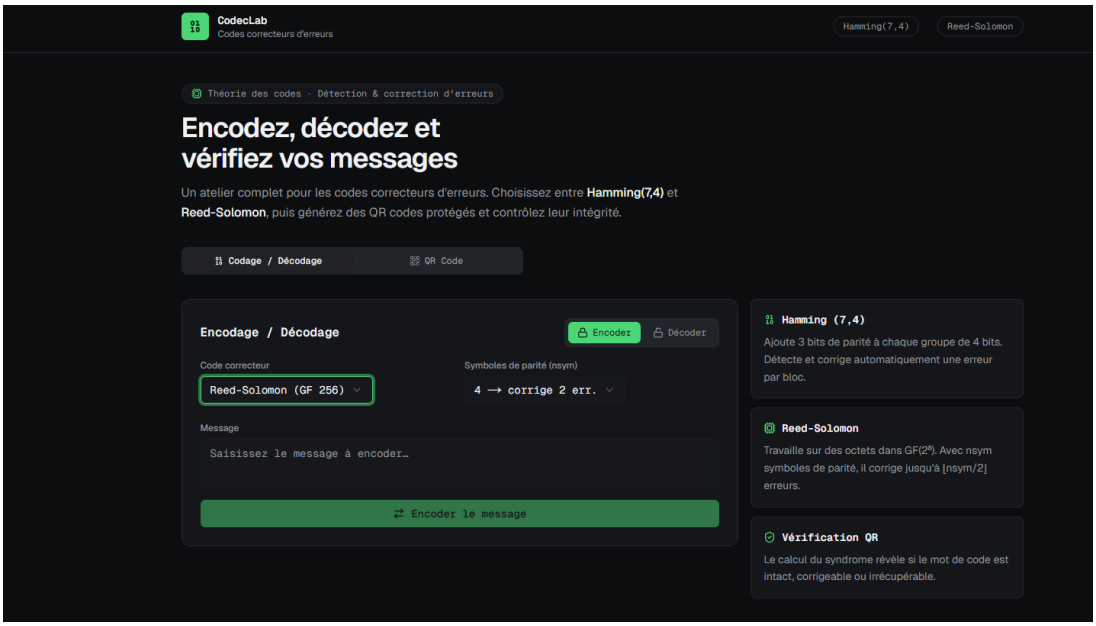

FIGURE 5.2 – Configuration du code Reed-Solomon (GF 256) avec réglage du nombre de symboles de parité n_{sym}.

Deux modes sont proposés via des boutons dédiés :

— **Encoder** : l'utilisateur saisit un message en clair dans une zone de texte, puis obtient le mot de code correspondant (bits de parité calculés pour Hamming, symboles de redondance calculés pour Reed-Solomon).

— **Décoder** : l'utilisateur colle un mot de code, éventuellement altéré par des erreurs, sous forme hexadécimale. L'application calcule le syndrome, détecte la présence d'erreurs, les localise et les corrige automatiquement avant d'afficher le message original retrouvé.

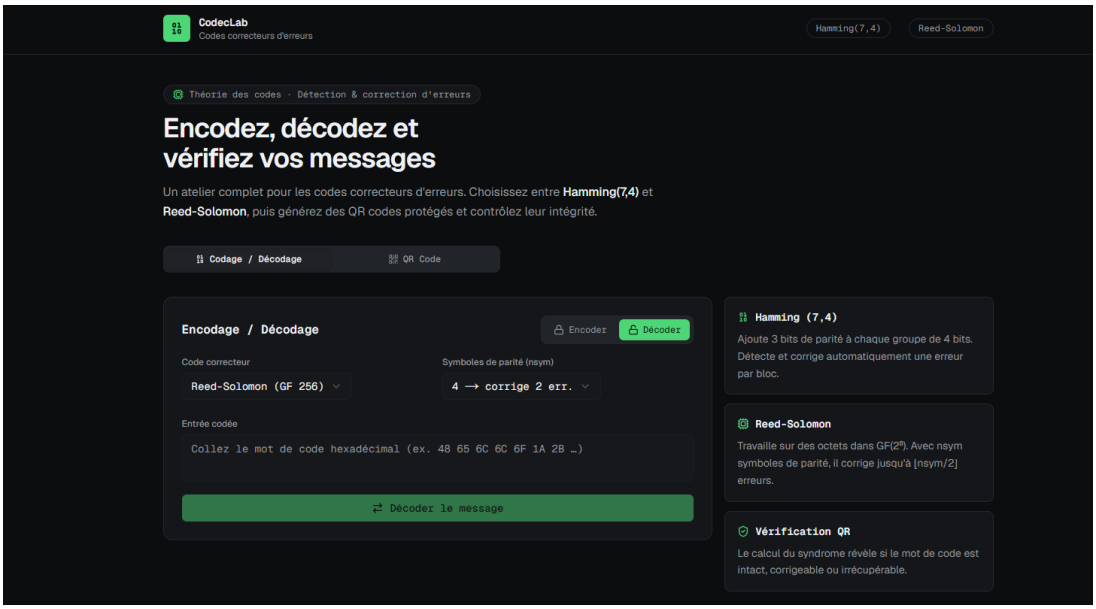

FIGURE 5.3 – Module de décodage : saisie d'un mot de code hexadécimal à corriger et décoder.

Module QR Code

Le second module illustre une application concrète et visuelle des codes correcteurs d'erreurs : la génération et la lecture de QR codes protégés. Il se compose de deux panneaux complémentaires.

— **Générer un QR code protégé** : l'utilisateur saisit soit un texte libre, soit un fichier / PDF, choisit le code correcteur (Hamming ou Reed-Solomon) ainsi que le niveau de parité souhaité, puis génère un QR code dans lequel le message est encodé avec la redondance correspondante.

— **Lire et vérifier un QR code** : l'utilisateur importe une ou plusieurs images de QR code (ou colle directement le contenu encodé, préfixé par `ECC:H74:` pour Hamming ou `ECC:RS:` pour Reed-Solomon). L'application calcule alors le syndrome du message reçu afin de déterminer si le contenu est intact, corrigeable, ou irrécupérable, puis restitue le message d'origine le cas échéant.

Ce module met en évidence l'intérêt pratique des codes correcteurs : un QR code partiellement endommagé (rayure, tache, pli) peut malgré tout être lu correctement grâce à la redondance introduite lors de l'encodage, exactement comme évoqué en section 5.3 pour les niveaux de correction normalisés des QR codes (7 %, 15 %, 25 %, 30 %).

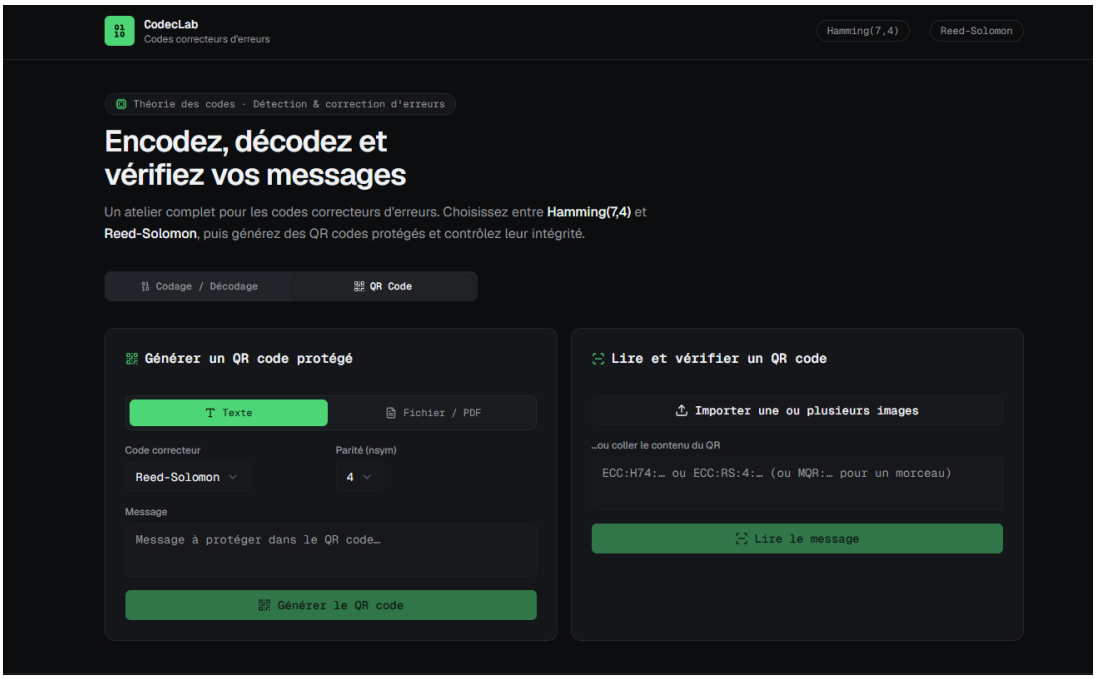

FIGURE 5.4 – Module QR Code : génération d'un QR code protégé et vérification/lecture d'un QR code existant.

Outil de développement

L'application CodecLab a été développée à l'aide de **v0**, la plateforme de génération d'interfaces web par intelligence artificielle éditée par **Vercel** (`https://v0.app`). Cet outil permet de concevoir rapidement des interfaces web modernes et fonctionnelles à partir de descriptions en langage naturel, puis de les déployer directement en ligne. L'ensemble de la logique de codage/décodage (Hamming, Reed-Solomon, génération et lecture de QR codes) a ensuite été implémenté et intégré à l'interface ainsi générée.

L'application est accessible en ligne à l'adresse suivante :
`https://vm-message-encoding-decoding.vusercontent.net/`

Conclusion

Ce projet a permis d'étudier deux codes correcteurs emblématiques, chacun illustrant une facette différente de la théorie des codes.

Le **code de Hamming (7,4)** se distingue par son élégance et sa minimalité : en ajoutant seulement 3 bits de parité, il atteint la perfection au sens de la borne de Hamming — ses boules de correction recouvrent exactement l'espace. Son décodage, fondé sur le calcul d'un syndrome et l'identification de la position d'erreur, est d'une simplicité remarquable. C'est un code idéal pour les situations où la ressource principale est la vitesse et où les erreurs sont rares et isolées.

Le **code de Reed-Solomon** représente une généralisation profonde. Travaillant sur des symboles (octets) plutôt que sur des bits, il exploite la richesse algébrique des corps de Galois pour atteindre la borne de Singleton — aucun code linéaire ne peut faire mieux avec les mêmes paramètres. Son décodage, bien que plus complexe (algorithme d'Euclide étendu, formule de Forney), reste polynomial et implémentable efficacement. Sa résistance aux erreurs en rafale en fait l'outil de référence pour la fiabilité des supports de stockage et des canaux de communication.

Les deux codes illustrent un principe fondamental : la **redondance structurée** transforme l'incertitude en certitude.

Perspectives

De nombreuses généralisations existent : les **codes BCH** (Bose-Chaudhuri-Hocquenghem) dont Reed-Solomon est un cas particulier ; les **codes LDPC** (faible densité) et les **Turbo codes** utilisés en 4G/5G ; les **codes d'effacement** pour le stockage distribué. Tous reposent sur les mêmes fondements algébriques explorés dans ce projet.

Bibliographie

Ouvrages de référence

[1] R. W. Hamming. *Error detecting and error correcting codes.* Bell System Technical Journal, vol. 29, no. 2, pp. 147–160, 1950.

[2] I. S. Reed, G. Solomon. *Polynomial Codes over Certain Finite Fields.* Journal of the Society for Industrial and Applied Mathematics (SIAM), vol. 8, no. 2, pp. 300–304, 1960.

[3] M. Demazure. *Cours d'Algèbre.* Éditions Cassini, 2008. Chapitres 9–10–11. ISBN 978-2-84225-127-7.

[4] J. H. van Lint. *Introduction to Coding Theory.* Springer-Verlag, 3ème édition, 1999. ISBN 978-3-540-64133-9.

[5] F. J. MacWilliams, N. J. A. Sloane. *The Theory of Error-Correcting Codes.* North-Holland Publishing Company, 1977.

Articles et ressources en ligne

[6] T. Hill. *Reed-Solomon Codes Explained.* BBC Research & Development, WHP031, mai 2013. `https://downloads.bbc.co.uk/rd/pubs/whp/whp-pdf-files/WHP031.pdf`

[7] B. Maggs. *Decoding Reed-Solomon Codes.* Duke Computer Science, cours avancés, octobre 2000.

[8] P. Shankar. *Decoding Reed-Solomon Codes Using Euclid's Algorithm.* Resonance — Journal of Science Education, avril 2007.

[9] J. Louis-Alexandre, C. Huynh, D. Lesbre. *Codes correcteurs d'erreur — TIPE.* Rapport académique, juin 2017.

[10] Wikipédia. *Code de Hamming, Code de Reed-Solomon, Corps de Galois.* `https://fr.wikipedia.org/wiki/Code_de_Hamming` (consulté 2025).

site pour la création de l'application

1. `https://v0.app`