Introduction aux Codes Correcteurs d'Erreurs

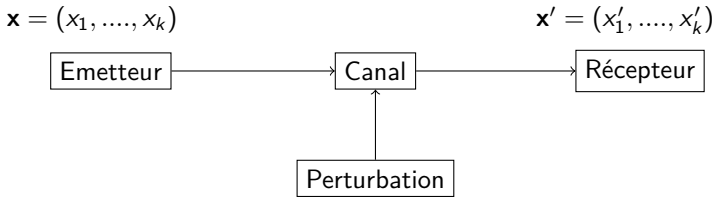
Hervé Talé Kalachi

Université de Yaoundé 1
(Maths/Info 3)

10 avril 2025

Plan

Canal de Transmission



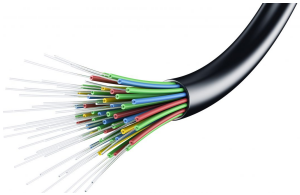
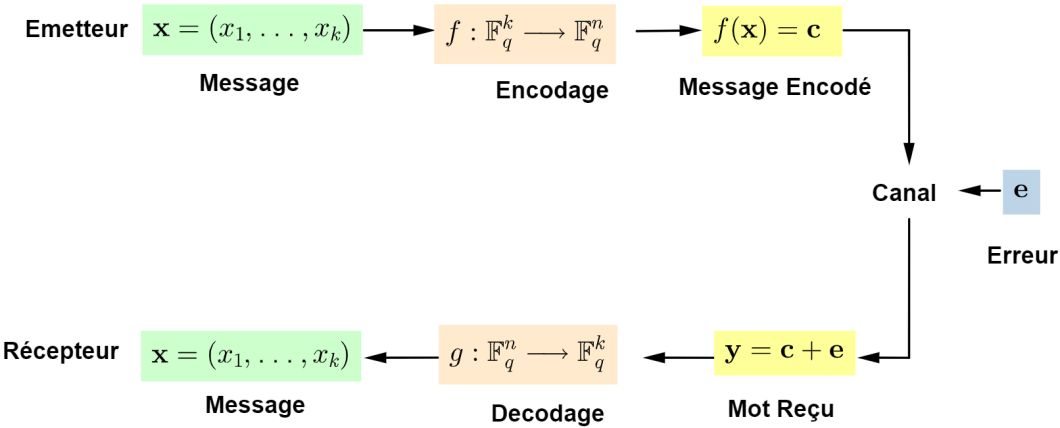
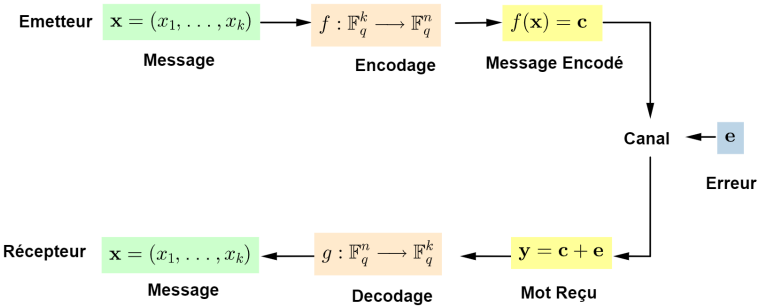


Schéma de la Théorie du Codage



Code : $\mathcal{C} = Im(f) \subset \mathbb{F}_q^n$

Schéma de la Théorie du Codage



Exemple 1 (Codes de Répétition)

Message : $\mathbf{x} = (x_1, x_2)$

Message Encodé : $\mathbf{c} = f(\mathbf{x}) = f(x_1, x_2) = (x_1, x_1, x_1, x_2, x_2, x_2)$

Mot Reçu : $\mathbf{y} = (x_1, x_1, x_1, x_2, x_1, x_2)$

Décodage : $\mathbf{y} = (x_1, x_1, x_1, x_2, x_1, x_2) \longrightarrow \mathbf{x} = (x_1, x_2)$

Prolèmes en Théorie du Codage

Codes à Répétition

Message : $\mathbf{x} = (x_1, x_2)$

Message Encodé : $\mathbf{c} = f(\mathbf{x}) = f(x_1, x_2) = (x_1, x_1, x_1, x_2, x_2, x_2)$

Problème : Capacité Correctrice

- S'il y a deux erreurs qui se produisent, est ce qu'on peut encore décoder ?
  $\mathbf{y} = (x_1, x_1, x_1, x_2, x_1, x_1) \longrightarrow \mathbf{x} = \_\ \_\ \_\ \_$?
- Quel est le nombre maximal d'erreurs qu'on peut décoder ?

Problème : Rendement

Le message utilise $k = 2$ symboles et le message encodé utilise $n = 6$ symboles.
Le rendement est de $\frac{k}{n} = \frac{1}{3}$.

- Est ce qu'on peut réduire le nombre de symboles utilisés pour encoder ?
- Est qu'on peut augmenter le rendement ?

Prolèmes en Théorie du Codage

Codes à Répétition

Message : $\mathbf{x} = (x_1, x_2)$

Message Encodé : $\mathbf{c} = f(\mathbf{x}) = f(x_1, x_2) = (x_1, x_1, x_1, x_2, x_2, x_2)$

Problème : Capacité Correctrice

- S'il y a deux erreurs qui se produisent, est ce qu'on peut encore décoder ?
  $\mathbf{y} = (x_1, x_1, x_1, x_2, x_1, x_1) \longrightarrow \mathbf{x} = \_\ \_\ \_\ \_$?
- Quel est le nombre maximal d'erreurs qu'on peut décoder ?

Problème : Rendement

Le message utilise $k = 2$ symboles et le message encodé utilise $n = 6$ symboles.
Le rendement est de $\frac{k}{n} = \frac{1}{3}$.

- Est ce qu'on peut réduire le nombre de symboles utilisés pour encoder ?
- Est qu'on peut augmenter le rendement ?

Prolèmes en Théorie du Codage

Codes à Répétition

Message : $\mathbf{x} = (x_1, x_2)$

Message Encodé : $\mathbf{c} = f(\mathbf{x}) = f(x_1, x_2) = (x_1, x_1, x_1, x_2, x_2, x_2)$

Problème : Capacité Correctrice

- S'il y a deux erreurs qui se produisent, est ce qu'on peut encore décoder ?
  $\mathbf{y} = (x_1, x_1, x_1, x_2, x_1, x_1) \longrightarrow \mathbf{x} = \_\ \_\ \_\ \_$?
- Quel est le nombre maximal d'erreurs qu'on peut décoder ?

Problème : Rendement

Le message utilise $k = 2$ symboles et le message encodé utilise $n = 6$ symboles.
Le rendement est de $\frac{k}{n} = \frac{1}{3}$.

- Est ce qu'on peut réduire le nombre de symboles utilisés pour encoder ?
- Est qu'on peut augmenter le rendement ?

Prolèmes en Théorie du Codage

Codes à Répétition

Message : $\mathbf{x} = (x_1, x_2)$

Message Encodé : $\mathbf{c} = f(\mathbf{x}) = f(x_1, x_2) = (x_1, x_1, x_1, x_2, x_2, x_2)$

Problème : Capacité Correctrice

- S'il y a deux erreurs qui se produisent, est ce qu'on peut encore décoder ?
  $\mathbf{y} = (x_1, x_1, x_1, x_2, x_1, x_1) \longrightarrow \mathbf{x} = \_\ \_\ \_\ \_$?
- Quel est le nombre maximal d'erreurs qu'on peut décoder ?

Problème : Rendement

Le message utilise $k = 2$ symboles et le message encodé utilise $n = 6$ symboles.
Le rendement est de $\frac{k}{n} = \frac{1}{3}$.

- Est ce qu'on peut réduire le nombre de symboles utilisés pour encoder ?
- Est qu'on peut augmenter le rendement ?

Prolèmes en Théorie du Codage

Codes à Répétition

Message : $\mathbf{x} = (x_1, x_2)$

Message Encodé : $\mathbf{c} = f(\mathbf{x}) = f(x_1, x_2) = (x_1, x_1, x_1, x_2, x_2, x_2)$

Problème : Capacité Correctrice

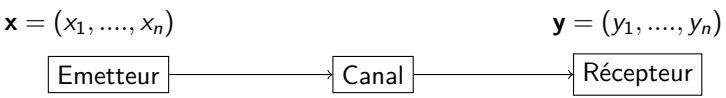
- S'il y a deux erreurs qui se produisent, est ce qu'on peut encore décoder ?
  $\mathbf{y} = (x_1, x_1, x_1, x_2, x_1, x_1) \longrightarrow \mathbf{x} = \_\ \_\ \_\ \_ ?$
- Quel est le nombre maximal d'erreurs qu'on peut décoder ?

Problème : Rendement

Le message utilise $k = 2$ symboles et le message encodé utilise $n = 6$ symboles.
Le rendement est de $\frac{k}{n} = \frac{1}{3}$.

- Est ce qu'on peut réduire le nombre de symboles utilisés pour encoder ?
- Est qu'on peut augmenter le rendement ?

Distance de Hamming



Distance de Hamming : $d_H(\mathbf{x}, \mathbf{y}) = Card\{i \ / \ x_i \neq y_i\}$.

Exemple 2

Mot transmis : $\mathbf{c} = (x_1, x_1, x_1, x_2, x_2, x_2)$ *Mot reçu :* $\mathbf{y} = (x_1, x_1, x_1, x_2, x_1, x_2)$

Nombre d'erreurs : $d_H(\mathbf{c}, \mathbf{y}) = 1$

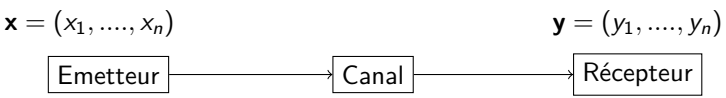
Proposition 3

L'application $d_H : \mathbb{F}_q^n \times \mathbb{F}_q^n \longrightarrow \mathbb{N}$ *est une distance :*

1) $d_H(\mathbf{x}, \mathbf{y}) \geqslant 0$; *2)* $d_H(\mathbf{x}, \mathbf{y}) = 0 \Longleftrightarrow \mathbf{x} = \mathbf{y}$

3) $d_H(\mathbf{x}, \mathbf{y}) = d_H(\mathbf{y}, \mathbf{x})$; *4)* $d_H(\mathbf{x}, \mathbf{z}) \leqslant d_H(\mathbf{x}, \mathbf{y}) + d_H(\mathbf{y}, \mathbf{z})$

Distance de Hamming



Distance de Hamming : $d_H(\mathbf{x}, \mathbf{y}) = Card\{i \ / \ x_i \neq y_i\}$.

Exemple 2

Mot transmis : $\mathbf{c} = (x_1, x_1, x_1, x_2, x_2, x_2)$ *Mot reçu :* $\mathbf{y} = (x_1, x_1, x_1, x_2, x_1, x_2)$

Nombre d'erreurs : $d_H(\mathbf{c}, \mathbf{y}) = 1$

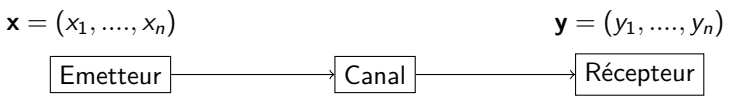
Proposition 3

L'application $d_H : \mathbb{F}_q^n \times \mathbb{F}_q^n \longrightarrow \mathbb{N}$ *est une distance :*

1) $d_H(\mathbf{x}, \mathbf{y}) \geqslant 0$; *2)* $d_H(\mathbf{x}, \mathbf{y}) = 0 \Longleftrightarrow \mathbf{x} = \mathbf{y}$

3) $d_H(\mathbf{x}, \mathbf{y}) = d_H(\mathbf{y}, \mathbf{x})$; *4)* $d_H(\mathbf{x}, \mathbf{z}) \leqslant d_H(\mathbf{x}, \mathbf{y}) + d_H(\mathbf{y}, \mathbf{z})$

Distance de Hamming



Distance de Hamming : $d_H(\mathbf{x}, \mathbf{y}) = Card\{i \ / \ x_i \neq y_i\}$.

Exemple 2

Mot transmis : $\mathbf{c} = (x_1, x_1, x_1, x_2, x_2, x_2)$ *Mot reçu :* $\mathbf{y} = (x_1, x_1, x_1, x_2, x_1, x_2)$

Nombre d'erreurs : $d_H(\mathbf{c}, \mathbf{y}) = 1$

Proposition 3

L'application $d_H : \mathbb{F}_q^n \times \mathbb{F}_q^n \longrightarrow \mathbb{N}$ *est une distance :*

1) $d_H(\mathbf{x}, \mathbf{y}) \geqslant 0$; *2)* $d_H(\mathbf{x}, \mathbf{y}) = 0 \Longleftrightarrow \mathbf{x} = \mathbf{y}$

3) $d_H(\mathbf{x}, \mathbf{y}) = d_H(\mathbf{y}, \mathbf{x})$; *4)* $d_H(\mathbf{x}, \mathbf{z}) \leqslant d_H(\mathbf{x}, \mathbf{y}) + d_H(\mathbf{y}, \mathbf{z})$

Distance Minimale et Capacité de Correction

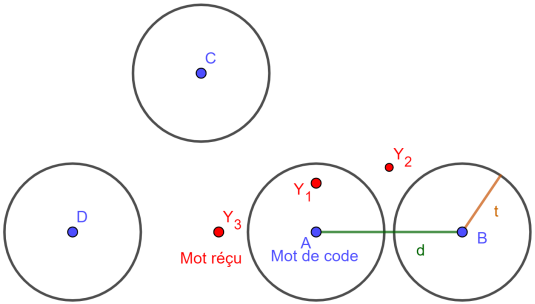
Definition 4

- $\mathcal{C}$ est un code sur $\mathbb{F}_q$
- La *distance minimale* d de $\mathcal{C}$ est :

$$d = \min\{d_H(\mathbf{x}, \mathbf{y}) : \ \mathbf{x}, \mathbf{y} \in \mathcal{C}, \mathbf{x} \neq \mathbf{y}\}.$$

- la *capacité de correction* t de $\mathcal{C}$ est la partie entière de $(d-1)/2$:

$$t = E(\frac{d-1}{2}).$$



Unicité du Décodage

La capacité de correction détermine le nombre d'erreurs que le code $\mathcal{C}$ peut corriger.

Proposition 5 (Unicité du Décodage)

Soit $\mathbf{y} \in \mathbb{F}_q^n$ un mot reçu d'un code $\mathcal{C}$ de capacité de correction t.

Alors, il existe un unique mot $\mathbf{c} \in \mathcal{C}$ tel que $d_H(\mathbf{y}, \mathbf{c}) \leqslant t$.

Preuve

Supposons qu'il existe $\mathbf{c}$ et $\mathbf{c}'$ dans $\mathcal{C}$ tels que $d_H(\mathbf{y}, \mathbf{c}) \leqslant t$ et $d_H(\mathbf{y}, \mathbf{c}') \leqslant t$. Alors,

$$\begin{aligned} d_H(\mathbf{c}, \mathbf{c}') &\leqslant d_H(\mathbf{c}, \mathbf{y}) + d_H(\mathbf{y}, \mathbf{c}') \\ &\leqslant t + t \\ &\leqslant 2E(\tfrac{d-1}{2}) \\ &\leqslant d - 1 \end{aligned}$$

Comme d est la distance minimale, alors $d_H(\mathbf{c}, \mathbf{c}') = 0$. D'où $\mathbf{c} = \mathbf{c}'$.

Unicité du Décodage

La capacité de correction détermine le nombre d'erreurs que le code $\mathcal{C}$ peut corriger.

Proposition 5 (Unicité du Décodage)

Soit $\mathbf{y} \in \mathbb{F}_q^n$ un mot reçu d'un code $\mathcal{C}$ de capacité de correction t.

Alors, il existe un unique mot $\mathbf{c} \in \mathcal{C}$ tel que $d_H(\mathbf{y}, \mathbf{c}) \leqslant t$.

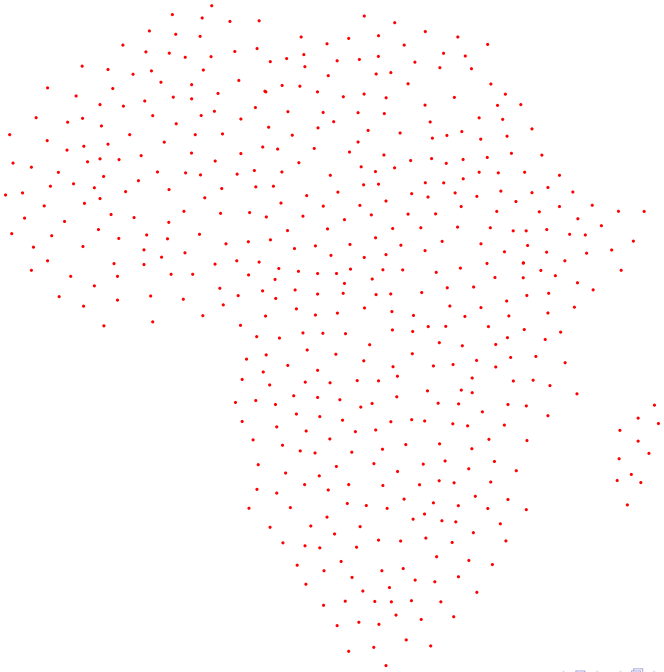
Preuve

Supposons qu'il existe $\mathbf{c}$ et $\mathbf{c}'$ dans $\mathcal{C}$ tels que $d_H(\mathbf{y}, \mathbf{c}) \leqslant t$ et $d_H(\mathbf{y}, \mathbf{c}') \leqslant t$. Alors,

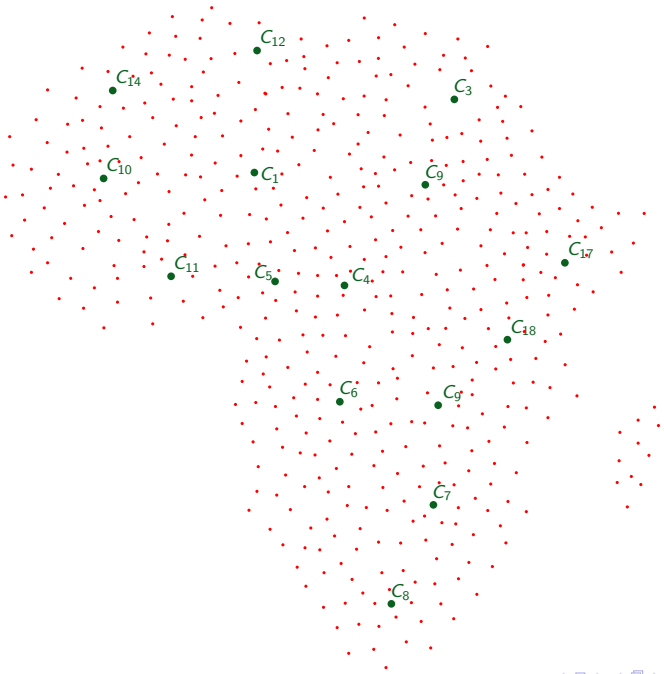
$$\begin{aligned} d_H(\mathbf{c}, \mathbf{c}') &\leqslant d_H(\mathbf{c}, \mathbf{y}) + d_H(\mathbf{y}, \mathbf{c}') \\ &\leqslant t + t \\ &\leqslant 2E(\tfrac{d-1}{2}) \\ &\leqslant d - 1 \end{aligned}$$

Comme d est la distance minimale, alors $d_H(\mathbf{c}, \mathbf{c}') = 0$. D'où $\mathbf{c} = \mathbf{c}'$. ■

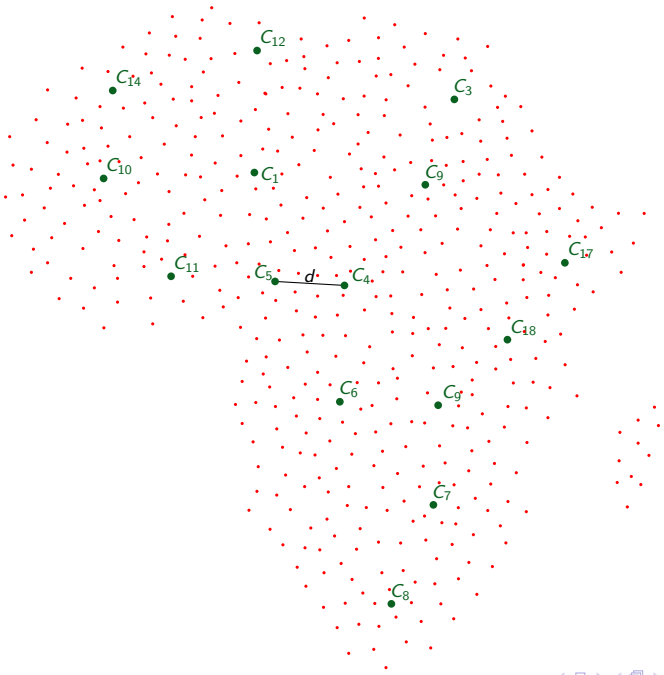
Distance Minimale et Capacité de Correction



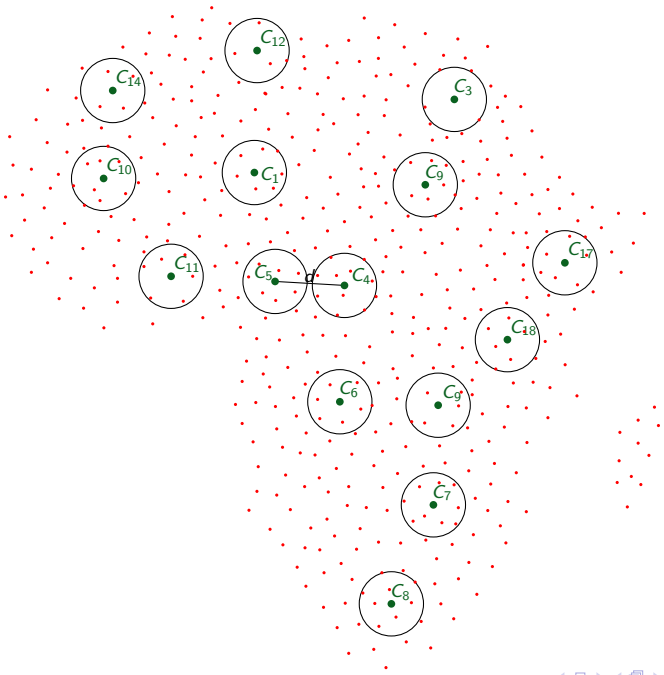
Distance Minimale et Capacité de Correction



Distance Minimale et Capacité de Correction



Distance Minimale et Capacité de Correction



Exemple

Considérons le code de répétition sur $\mathbb{F}_2 = \{0, 1\}$.

Message : $\mathbf{x} = (x_1, x_2) \in \mathbb{F}_2^2$.

Mot du code : $\mathbf{c} = f(\mathbf{x}) = (x_1, x_1, x_1, x_2, x_2, x_2)$

Messages	Mots du code
(0,0)	(0,0,0,0,0,0)
(0,1)	(0,0,0,1,1,1)
(1,0)	(1,1,1,0,0,0)
(1,1)	(1,1,1,1,1,1)

Exemple

		Distance de Hamming entre deux mots du code			
Messages	Mots du code	(0,0,0,0,0,0)	(0,0,0,1,1,1)	(1,1,1,0,0,0)	(1,1,1,1,1,1)
(0,0)	(0,0,0,0,0,0)				
(0,1)	(0,0,0,1,1,1)				
(1,0)	(1,1,1,0,0,0)				
(1,1)	(1,1,1,1,1,1)				

Exemple

		Distance de Hamming entre deux mots du code			
Messages	Mots du code	(0,0,0,0,0,0)	(0,0,0,1,1,1)	(1,1,1,0,0,0)	(1,1,1,1,1,1)
(0,0)	(0,0,0,0,0,0)	0	3	3	6
(0,1)	(0,0,0,1,1,1)	3	0	6	3
(1,0)	(1,1,1,0,0,0)	3	6	0	3
(1,1)	(1,1,1,1,1,1)	6	3	3	0

Exemple

		Distance de Hamming entre deux mots du code			
Messages	Mots du code	(0,0,0,0,0,0)	(0,0,0,1,1,1)	(1,1,1,0,0,0)	(1,1,1,1,1,1)
(0,0)	(0,0,0,0,0,0)	0	3	3	6
(0,1)	(0,0,0,1,1,1)	3	0	6	3
(1,0)	(1,1,1,0,0,0)	3	6	0	3
(1,1)	(1,1,1,1,1,1)	6	3	3	0

Donc la distance minimale est $d = 3$ et la capacité de correction est $t = E(\frac{d-1}{2}) = 1$.

Ainsi, le code de répétition peut corriger 1 erreur.

Plan

Definition 6

Codes Linéaires

1. Un **code linéaire** de longueur n et de dimension k est un espace vectoriel sur $\mathbb{F}_q$

$$\mathcal{C} = \bigoplus_{i=1}^{k} \mathbb{F}_q \, \mathbf{v}_i$$

où $\mathbf{v}_i \in \mathbb{F}_q^n$ sont linéairement indépendants.

2. Toute matrice $k \times n$, $\boldsymbol{G}$ dont les lignes forment une base de $\mathcal{C}$ est une matrice génératrice de $\mathcal{C}$.

$$\mathcal{C} = \{\mathbf{m}\boldsymbol{G},\ \mathbf{m} \in \mathbb{F}_q^k\}$$

Exemple 7 (Codes de Répétition)

$$\mathbf{x} = (x_1, x_2) \qquad \begin{aligned} \mathbf{c} = f(\mathbf{x}) = f(x_1, x_2) &= (x_1, x_1, x_1, x_2, x_2, x_2) \\ &= (x_1, x_2) \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix} = \mathbf{x}\mathbf{G} \end{aligned}$$

Definition 6

Codes Linéaires

1. Un **code linéaire** de longueur n et de dimension k est un espace vectoriel sur $\mathbb{F}_q$

$$\mathcal{C} = \bigoplus_{i=1}^{k} \mathbb{F}_q \, \mathbf{v}_i$$

où $\mathbf{v}_i \in \mathbb{F}_q^n$ sont linéairement indépendants.

2. Toute matrice $k \times n$, $\boldsymbol{G}$ dont les lignes forment une base de $\mathcal{C}$ est une matrice génératrice de $\mathcal{C}$.

$$\mathcal{C} = \{\mathbf{m}\boldsymbol{G},\ \mathbf{m} \in \mathbb{F}_q^k\}$$

Exemple 7 (Codes de Répétition)

$$\mathbf{x} = (x_1, x_2) \qquad \begin{aligned} \mathbf{c} = f(\mathbf{x}) = f(x_1, x_2) &= (x_1, x_1, x_1, x_2, x_2, x_2) \\ &= (x_1, x_2) \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix} = \mathbf{xG} \end{aligned}$$

Poids de Hamming

Definition 8

Poids de Hamming Le **poids de Hamming** d'un mot $\mathbf{x} \in \mathcal{C}$ est simplement sa distance au mot nul. On le notera $w(\mathbf{x})$ ou $w_H(\mathbf{x})$.

$$w_H(\mathbf{x}) = d_H(\mathbf{x}, \mathbf{0}) = card\{i \ / \ x_i \neq 0\}$$

Proposition 9

La distance minimale d d'un code linéaire $\mathcal{C}$ est le poids minimum des mots du code.

$$\begin{aligned} d &= min\{w(\mathbf{x}) : \mathbf{x} \in \mathcal{C}, \mathbf{x} \neq \mathbf{0}\} \\ &= min\{w(\mathbf{x} - \mathbf{y}) : \mathbf{x}, \mathbf{y} \in \mathcal{C}, \mathbf{x} \neq \mathbf{y}\} \\ &= min\{d_H(\mathbf{x}, \mathbf{y}) : \mathbf{x}, \mathbf{y} \in \mathcal{C}, \mathbf{x} \neq \mathbf{y}\} \end{aligned}$$

Notation 10

Désormais, un code linéaire sur $\mathbb{F}_q$ de longueur n, de dimension k et de distance minimale d sera simplement appelé code $[n, k, d]$ ou $[n, k, d]$ code.

Poids de Hamming

Definition 8

Poids de Hamming Le **poids de Hamming** d'un mot $\mathbf{x} \in \mathcal{C}$ est simplement sa distance au mot nul. On le notera $w(\mathbf{x})$ ou $w_H(\mathbf{x})$.

$$w_H(\mathbf{x}) = d_H(\mathbf{x}, \mathbf{0}) = card\{i \ / \ x_i \neq 0\}$$

Proposition 9

La distance minimale d d'un code linéaire $\mathcal{C}$ est le poids minimum des mots du code.

$$\begin{aligned} d &= min\{w(\mathbf{x}) : \mathbf{x} \in \mathcal{C}, \mathbf{x} \neq \mathbf{0}\} \\ &= min\{w(\mathbf{x} - \mathbf{y}) : \mathbf{x}, \mathbf{y} \in \mathcal{C}, \mathbf{x} \neq \mathbf{y}\} \\ &= min\{d_H(\mathbf{x}, \mathbf{y}) : \mathbf{x}, \mathbf{y} \in \mathcal{C}, \mathbf{x} \neq \mathbf{y}\} \end{aligned}$$

Notation 10

Désormais, un code linéaire sur $\mathbb{F}_q$ de longueur n, de dimension k et de distance minimale d sera simplement appelé code $[n, k, d]$ ou $[n, k, d]$ code.

Poids de Hamming

Definition 8

Poids de Hamming Le **poids de Hamming** d'un mot $\mathbf{x} \in \mathcal{C}$ est simplement sa distance au mot nul. On le notera $w(\mathbf{x})$ ou $w_H(\mathbf{x})$.

$$w_H(\mathbf{x}) = d_H(\mathbf{x}, \mathbf{0}) = card\{i \ / \ x_i \neq 0\}$$

Proposition 9

La distance minimale d d'un code linéaire $\mathcal{C}$ est le poids minimum des mots du code.

$$\begin{aligned} d &= min\{w(\mathbf{x}) : \mathbf{x} \in \mathcal{C}, \mathbf{x} \neq \mathbf{0}\} \\ &= min\{w(\mathbf{x} - \mathbf{y}) : \mathbf{x}, \mathbf{y} \in \mathcal{C}, \mathbf{x} \neq \mathbf{y}\} \\ &= min\{d_H(\mathbf{x}, \mathbf{y}) : \mathbf{x}, \mathbf{y} \in \mathcal{C}, \mathbf{x} \neq \mathbf{y}\} \end{aligned}$$

Notation 10

Désormais, un code linéaire sur $\mathbb{F}_q$ de longueur n, de dimension k et de distance minimale d sera simplement appelé code $[n, k, d]$ ou $[n, k, d]$ code.

Borne de Singleton

Théorème 11 (*Borne de Singleton*)

Si $\mathcal{C}$ est $(n, k, d)-$ code, alors

$$d \leqslant n - k + 1$$

Definition 12 (MDS Code)

An $(n, k, d)-$code $\mathcal{C}$ is said to be MDS (Maximum Distance Separable) if the singleton bound is reached. That is to say :

$$d = n - k + 1$$

Borne de Singleton

Théorème 11 (*Borne de Singleton*)

Si $\mathcal{C}$ est $(n, k, d)-$ code, alors

$$d \leqslant n - k + 1$$

Definition 12 (MDS Code)

An $(n, k, d)-$code $\mathcal{C}$ is said to be MDS (Maximum Distance Separable) if the singleton bound is reached. That is to say :

$$d = n - k + 1$$

Code Dual

Definition 13 (Produit Scalaire)

Soient $\mathbf{x} = (x_1, ..., x_n)$ et $\mathbf{y} = (y_1, ..., y_n)$ deux éléments de $\mathbb{F}_q^n$.

- On appelle produit scalaire de $\mathbf{x}$ par $\mathbf{y}$ noté $\mathbf{x} \cdot \mathbf{y}$ la quantité

$$\mathbf{x} \cdot \mathbf{y} = \mathbf{x}\mathbf{y}^T = (x_1, \ldots, x_n) \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} = \sum_{i=1}^{n} x_i \cdot y_i.$$

- $\mathbf{x}$ est orthogonal à $\mathbf{y}$ si et seulement si $\mathbf{x} \cdot \mathbf{y} = \mathbf{0}$.

Definition 14

Soit $\mathcal{C}$ un code $[n, k]$ sur $\mathbb{F}_q$. On appelle orthogonal (ou dual) de $\mathcal{C}$ et on note $\mathcal{C}^\perp$, le sous-espace vectoriel orthogonal à $\mathcal{C}$; c'est-à-dire :

$$\mathcal{C}^\perp = \left\{ \mathbf{y} \in \mathbb{F}_q^n \mid \mathbf{x} \cdot \mathbf{y} = 0, \forall \mathbf{x} \in \mathcal{C} \right\}$$

Code Dual

Definition 13 (Produit Scalaire)

Soient $\mathbf{x} = (x_1, ..., x_n)$ et $\mathbf{y} = (y_1, ..., y_n)$ deux éléments de $\mathbb{F}_q^n$.

- On appelle produit scalaire de $\mathbf{x}$ par $\mathbf{y}$ noté $\mathbf{x} \cdot \mathbf{y}$ la quantité

$$\mathbf{x} \cdot \mathbf{y} = \mathbf{x}\mathbf{y}^T = (x_1, \ldots, x_n) \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} = \sum_{i=1}^{n} x_i \cdot y_i.$$

- $\mathbf{x}$ est orthogonal à $\mathbf{y}$ si et seulement si $\mathbf{x} \cdot \mathbf{y} = \mathbf{0}$.

Definition 14

Soit $\mathcal{C}$ un code $[n, k]$ sur $\mathbb{F}_q$. On appelle orthogonal (ou dual) de $\mathcal{C}$ et on note $\mathcal{C}^\perp$, le sous-espace vectoriel orthogonal à $\mathcal{C}$; c'est-à-dire :

$$\mathcal{C}^\perp = \left\{ \mathbf{y} \in \mathbb{F}_q^n \mid \mathbf{x} \cdot \mathbf{y} = 0, \forall \mathbf{x} \in \mathcal{C} \right\}$$

Code Dual

Proposition 15

Soit $\mathcal{C}$ un code $[n, k]$ de matrice génératrice $\boldsymbol{G}$.

(i) $\mathcal{C}^{\perp} = \left\{ \mathbf{y} \in \mathbb{F}_q^n \mid \boldsymbol{G}\mathbf{y}^T = 0 \right\}$ *; où* $\mathbf{y}^T$ *est la transposée de* $\mathbf{y}$.

(ii) $\mathcal{C}^{\perp}$ *est un code* $[n, n-k]$.

(iii) $(\mathcal{C}^{\perp})^{\perp} = \mathcal{C}$.

Matrice de Contrôle

Definition 16 (Matrice de Contrôle)

On appelle **matrice de contrôle** (ou de parité) d'un $[n, k]$ code $\mathcal{C}$, toute matrice génératrice de son orthogonal $\mathcal{C}^{\perp}$.

Proposition 17

Soit $\mathcal{C}$ un code $[n, k]$ de matrice génératrice $\boldsymbol{G}$. Soit $\boldsymbol{H}$ une matrice $(n-k) \times n$ de rang $n-k$. Alors les propriétés suivantes sont équivalentes :

(i) *$\boldsymbol{H}$ est une matrice de contrôle de $\mathcal{C}$*

(ii) *$\mathbf{x} \in \mathcal{C}$ si et seulement si $\boldsymbol{H} \cdot \mathbf{x}^t = \mathbf{0}$*

(iii) *$\boldsymbol{G}\boldsymbol{H}^T = \mathbf{0}$.*

Definition 18 (Syndrome)

Soit $\boldsymbol{H}$ une matrice de contrôle d'un code linéaire $\mathcal{C}$. On appelle **syndrome** de $\mathbf{x} \in \mathbb{F}_q^n$ l'élément $\mathbf{s} = \mathbf{x}\boldsymbol{H}^T$

Matrice de Contrôle

Definition 16 (Matrice de Contrôle)

On appelle **matrice de contrôle** (ou de parité) d'un $[n, k]$ code $\mathcal{C}$, toute matrice génératrice de son orthogonal $\mathcal{C}^{\perp}$.

Proposition 17

Soit $\mathcal{C}$ un code $[n, k]$ de matrice génératrice $\boldsymbol{G}$. Soit $\boldsymbol{H}$ une matrice $(n-k) \times n$ de rang $n-k$. Alors les propriétés suivantes sont équivalentes :

(i) $\boldsymbol{H}$ *est une matrice de contrôle de* $\mathcal{C}$

(ii) $\mathbf{x} \in \mathcal{C}$ *si et seulement si* $\boldsymbol{H} \cdot \mathbf{x}^t = \mathbf{0}$

(iii) $\boldsymbol{G}\boldsymbol{H}^T = \mathbf{0}$.

Definition 18 (Syndrome)

Soit $\boldsymbol{H}$ une matrice de contrôle d'un code linéaire $\mathcal{C}$. On appelle **syndrome** de $\mathbf{x} \in \mathbb{F}_q^n$ l'élément $\mathbf{s} = \mathbf{x}\boldsymbol{H}^T$

Matrice de Contrôle

Definition 16 (Matrice de Contrôle)

On appelle **matrice de contrôle** (ou de parité) d'un $[n, k]$ code $\mathcal{C}$, toute matrice génératrice de son orthogonal $\mathcal{C}^{\perp}$.

Proposition 17

Soit $\mathcal{C}$ un code $[n, k]$ de matrice génératrice $\boldsymbol{G}$. Soit $\boldsymbol{H}$ une matrice $(n-k) \times n$ de rang $n-k$. Alors les propriétés suivantes sont équivalentes :

(i) $\boldsymbol{H}$ *est une matrice de contrôle de* $\mathcal{C}$

(ii) $\mathbf{x} \in \mathcal{C}$ *si et seulement si* $\boldsymbol{H} \cdot \mathbf{x}^t = \mathbf{0}$

(iii) $\boldsymbol{G}\boldsymbol{H}^T = \mathbf{0}$.

Definition 18 (Syndrome)

Soit $\boldsymbol{H}$ une matrice de contrôle d'un code linéaire $\mathcal{C}$. On appelle **syndrome** de $\mathbf{x} \in \mathbb{F}_q^n$ l'élément $\mathbf{s} = \mathbf{x}\boldsymbol{H}^T$

Forme Systématique

Proposition 19

Soit $\mathcal{C}$ un code $[n, k]$ linéaire de matrice génératrice $\mathbf{G}$. Alors,

(a) *Il existe une matrice inversible S et une matrice de permutation $\mathbf{P}$ telles que :*

$$\mathbf{G} = \mathbf{S} \begin{pmatrix} \mathbf{I}_k & \mathbf{A} \end{pmatrix} \mathbf{P}.$$

(b) *Une matrice de contrôle de $\mathcal{C}$ est :*

$$\mathbf{H} = \begin{pmatrix} -\mathbf{A} & \mathbf{I}_{n-k} \end{pmatrix} \mathbf{P}^{-1} = \begin{pmatrix} -\mathbf{A}^T & \mathbf{I}_{n-k} \end{pmatrix} \mathbf{P}^T.$$

Exemple

La matrice génératrice du code de répétition de l'exemple précédent est

$$\mathbf{G} = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \end{pmatrix} \mathbf{P} = \begin{pmatrix} \mathbf{I}_2 & \mathbf{A} \end{pmatrix} \mathbf{P}$$

où

$$\mathbf{P} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Donc la matrice de contrôle de $\mathcal{C}$ est

$$\mathbf{H} = \begin{pmatrix} -\mathbf{A}^T & \mathbf{I}_{n-k} \end{pmatrix} \mathbf{P}^T = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \mathbf{P}^T = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix}$$

Exemple

La matrice génératrice du code de répétition de l'exemple précédent est

$$\mathbf{G} = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \end{pmatrix} \mathbf{P} = \begin{pmatrix} \mathbf{I}_2 & \mathbf{A} \end{pmatrix} \mathbf{P}$$

où

$$\mathbf{P} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Donc la matrice de contrôle de $\mathcal{C}$ est

$$\mathbf{H} = \begin{pmatrix} -\mathbf{A}^T & \mathbf{I}_{n-k} \end{pmatrix} \mathbf{P}^T = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \mathbf{P}^T = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix}$$

Plan

Codes de Reed-Solomon

Definition 20

Le **code de Reed-Solomon** $RS_k(\mathbf{L})$ de longueur n et de dimension k est :

$$RS_k(\mathbf{a}) = \{((f(a_1), ..., f(a_n)) \ / \ f \in \mathbb{F}_q[X] \ , deg(f) < k\}$$

$\mathbf{a}$ est appelé le support de $RS_k(\mathbf{a})$.

Proposition 21

(i) *$RS_k(\boldsymbol{a})$ est un code linéaire de longueur n et de dimension k.*

(ii) *Une matrice génératrice de $RS_k(\boldsymbol{a})$ est :*

$$\boldsymbol{G} = \begin{pmatrix} 1 & 1 & . & . & . & . & 1 \\ a_1 & a_2 & . & . & . & . & a_n \\ a_1^2 & a_2^2 & . & . & . & . & a_n^2 \\ . & . & & & & & . \\ . & . & & & & & . \\ . & . & & & & & . \\ a_1^{k-1} & a_2^{k-1} & . & . & . & . & a_n^{k-1} \end{pmatrix}$$

(iii) *$RS_k(\boldsymbol{a})$ est un code MDS.*

Codes de Reed-Solomon

- k et n sont deux entiers tels que $1 \leqslant k \leqslant n$.
- $\mathbf{a} = (a_1, ..., a_n)$ tel que $a_i \in \mathbb{F}_q$ et les a_i deux à deux distincts.

Message : $\mathbf{m} = (m_0, m_1, ..., m_{k-1})$.

Polynôme associé au message : $f(X) = m_0 + m_1 X + ... + m_{k-1} X^{k-1}$.

Encodage : $\mathbf{c} = (f(a_1), f(a_2), ..., f(a_n))$

$$\mathbf{c} = (m_0 + m_1 a_1 + ... + m_{k-1} {a_1}^{k-1}, m_0 + m_1 a_2 + ... + m_{k-1} {a_2}^{k-1}, ..., m_0 + m_1 a_n + ... + m_{k-1} {a_n}^{k-1})$$

$$\mathbf{c} = (m_0, ..., m_{k-1}) \begin{pmatrix} 1 & 1 & . & . & . & . & 1 \\ a_1 & a_2 & . & . & . & . & a_n \\ {a_1}^2 & {a_2}^2 & . & . & . & . & {a_n}^2 \\ . & . & & & & & . \\ . & . & & & & & . \\ . & . & & & & & . \\ {a_1}^{k-1} & {a_2}^{k-1} & . & . & . & . & {a_n}^{k-1} \end{pmatrix}$$

Décodage : Algorithme de Berlekamp-Welch

- Mot de code : $\mathbf{c} = (\mathcal{P}_c(a_1), \mathcal{P}_c(a_2), ..., \mathcal{P}_c(a_n))$, où $\mathcal{P}_c \in \mathbb{F}_q[X]$, $deg(\mathcal{P}_c) < k$.
- Mot reçu : $\mathbf{y} = \mathbf{c} + \mathbf{e}$, où $\mathbf{e}$ est une erreur de poids w.
- **Polynôme Localisateur** de l'erreur $\mathbf{e}$ est le polynôme unitaire $\mathcal{L}_e$ (de degré minimal $= w$), tel que :

$$\mathcal{L}_e(a_i) = 0 \quad \text{si} \quad e_i \neq 0$$

On a $\forall i \in [|1; n|]$:

$$\begin{aligned} y_i \times \mathcal{L}_e(a_i) &= (\mathcal{P}_c(a_i) + e_i) \times \mathcal{L}_e(a_i) \\ &= \mathcal{P}_c(a_i) \times \mathcal{L}_e(a_i) \\ &= \mathcal{N}(a_i). \end{aligned}$$

Avec $\mathcal{N} = \mathcal{P}_c \times \mathcal{L}_e$ qui est donc de degré au plus $k - 1 + w$.

Algorithme de Berlekamp-Welch

On obtient alors le système linéaire suivant :

$$\forall i \in [|1; n|] \quad \mathbf{y}\mathcal{L}_e(a_i) = \mathcal{N}(a_i).$$

Il est composé de

- n équations
- $k + 2w$ inconnues
  - w inconnues pour $\mathcal{L}_e$
  - $k + w$ pour $\mathcal{N}$

Algorithme de Berlekamp-Welch

On obtient alors le système linéaire suivant :

$$\forall i \in [|1; n|] \quad \mathbf{y}\mathcal{L}_e(a_i) = \mathcal{N}(a_i).$$

Il est composé de

- n équations
- $k + 2w$ inconnues
  - w inconnues pour $\mathcal{L}_e$
  - $k + w$ pour $\mathcal{N}$

Exemple

- Sur $\mathbb{F}_5 = \{0, 1, 2, 3, 4\}$,
- Considérons le $[4, 2, 3]$ code $RS_2(\mathbf{a})$ avec $\mathbf{a} = (1, 2, 4, 3)$.
- Message : $\mathbf{m} = (2, 3)$.
- Polynôme associé au message : $f(X) = 2 + 3$.
- Encodage : $\mathbf{c} = (f(1), f(2), f(4), f(3) = \mathbf{c} = (0, 3, 4, 1)$
- Erreur : $\mathbf{e} = (0, 0, 1, 0)$
- Mot reçu : $\mathbf{e} = \mathbf{y} = \mathbf{c} + \mathbf{e} = (0, 3, 0, 1)$

Exemple

$\mathbf{a} = (1, 2, 4, 3)$.

Nous allons décoder $\mathbf{y} = (0, 3, 0, 1)$. On a :

- $\forall i \in [|1; 4|] \quad y_i \times \mathcal{L}_e(a_i) = N(a_i)$.
- De plus $deg(N) = 2$ et $deg(\mathcal{L}_e) = 1$; d'où $N(x) = ax^2 + bx + c$ et $\mathcal{L}_e(x) = x + d$.
- On a alors le système suivant :

$$\begin{cases} a + b + c = 0 \\ 4a + 2b + c = 3(2 + d) \\ a + 4b + c = 0 \\ 4a + 3b + c = 3 + d \end{cases}$$

- D'où

$$\begin{cases} a + b + c = 0 \\ 4a + 2b + c - 3d = 1 \\ a + 4b + c = 0 \\ 4a + 3b + c - d = 3 \end{cases}$$

- Ce qui nous permet d'obtenir après résolution $a = 3$, $b = 0$, $c = 2$ et $d = 1$.
- Donc $N(x) = 3x^2 + 2 = f(x)(x + 1)$. Par une division Euclidienne, on a $f(x) = 3x + 3$.

Exemple

$\mathbf{a} = (1, 2, 4, 3)$.

Nous allons décoder $\mathbf{y} = (0, 3, 0, 1)$. On a :

- $\forall i \in [|1; 4|] \quad y_i \times \mathcal{L}_e(a_i) = N(a_i)$.
- De plus $deg(N) = 2$ et $deg(\mathcal{L}_e) = 1$; d'où $N(x) = ax^2 + bx + c$ et $\mathcal{L}_e(x) = x + d$.
- On a alors le système suivant :

$$\begin{cases} a + b + c = 0 \\ 4a + 2b + c = 3(2 + d) \\ a + 4b + c = 0 \\ 4a + 3b + c = 3 + d \end{cases}$$

- D'où

$$\begin{cases} a + b + c = 0 \\ 4a + 2b + c - 3d = 1 \\ a + 4b + c = 0 \\ 4a + 3b + c - d = 3 \end{cases}$$

- Ce qui nous permet d'obtenir après résolution $a = 3$, $b = 0$, $c = 2$ et $d = 1$.
- Donc $N(x) = 3x^2 + 2 = f(x)(x + 1)$. Par une division Euclidienne, on a $f(x) = 3x + 2$.

Exemple

$\mathbf{a} = (1, 2, 4, 3)$.

Nous allons décoder $\mathbf{y} = (0, 3, 0, 1)$. On a :

- $\forall i \in [|1; 4|] \quad y_i \times \mathcal{L}_e(a_i) = N(a_i)$.
- De plus $deg(N) = 2$ et $deg(\mathcal{L}_e) = 1$; d'où $N(x) = ax^2 + bx + c$ et $\mathcal{L}_e(x) = x + d$.
- On a alors le système suivant :

$$\begin{cases} a + b + c = 0 \\ 4a + 2b + c = 3(2 + d) \\ a + 4b + c = 0 \\ 4a + 3b + c = 3 + d \end{cases}$$

- D'où

$$\begin{cases} a + b + c = 0 \\ 4a + 2b + c - 3d = 1 \\ a + 4b + c = 0 \\ 4a + 3b + c - d = 3 \end{cases}$$

- Ce qui nous permet d'obtenir après résolution $a = 3$, $b = 0$, $c = 2$ et $d = 1$.
- Donc $N(x) = 3x^2 + 2 = f(x)(x + 1)$. Par une division Euclidienne, on a $f(x) = 3x + 2$.

Exemple

$\mathbf{a} = (1, 2, 4, 3)$.

Nous allons décoder $\mathbf{y} = (0, 3, 0, 1)$. On a :

- $\forall i \in [|1; 4|] \quad y_i \times \mathcal{L}_e(a_i) = N(a_i)$.
- De plus $deg(N) = 2$ et $deg(\mathcal{L}_e) = 1$; d'où $N(x) = ax^2 + bx + c$ et $\mathcal{L}_e(x) = x + d$.
- On a alors le système suivant :

$$\begin{cases} a + b + c = 0 \\ 4a + 2b + c = 3(2 + d) \\ a + 4b + c = 0 \\ 4a + 3b + c = 3 + d \end{cases}.$$

- D'où

$$\begin{cases} a + b + c = 0 \\ 4a + 2b + c - 3d = 1 \\ a + 4b + c = 0 \\ 4a + 3b + c - d = 3 \end{cases}.$$

- Ce qui nous permet d'obtenir après résolution $a = 3$, $b = 0$, $c = 2$ et $d = 1$.
- Donc $N(x) = 3x^2 + 2 = f(x)(x + 1)$. Par une division Euclidienne, on a $f(x) = 3x + 2$.

Exemple

$\mathbf{a} = (1, 2, 4, 3)$.

Nous allons décoder $\mathbf{y} = (0, 3, 0, 1)$. On a :

- $\forall i \in [|1; 4|] \quad y_i \times \mathcal{L}_e(a_i) = N(a_i)$.
- De plus $deg(N) = 2$ et $deg(\mathcal{L}_e) = 1$; d'où $N(x) = ax^2 + bx + c$ et $\mathcal{L}_e(x) = x + d$.
- On a alors le système suivant :

$$\begin{cases} a + b + c = 0 \\ 4a + 2b + c = 3(2 + d) \\ a + 4b + c = 0 \\ 4a + 3b + c = 3 + d \end{cases}.$$

- D'où

$$\begin{cases} a + b + c = 0 \\ 4a + 2b + c - 3d = 1 \\ a + 4b + c = 0 \\ 4a + 3b + c - d = 3 \end{cases}.$$

- Ce qui nous permet d'obtenir après résolution $a = 3$, $b = 0$, $c = 2$ et $d = 1$.
- Donc $N(x) = 3x^2 + 2 = f(x)(x + 1)$. Par une division Euclidienne, on a $f(x) = 3x + 2$.

Exemple

$\mathbf{a} = (1, 2, 4, 3)$.

Nous allons décoder $\mathbf{y} = (0, 3, 0, 1)$. On a :

- $\forall i \in [|1; 4|] \quad y_i \times \mathcal{L}_e(a_i) = N(a_i)$.
- De plus $deg(N) = 2$ et $deg(\mathcal{L}_e) = 1$; d'où $N(x) = ax^2 + bx + c$ et $\mathcal{L}_e(x) = x + d$.
- On a alors le système suivant :

$$\begin{cases} a + b + c = 0 \\ 4a + 2b + c = 3(2 + d) \\ a + 4b + c = 0 \\ 4a + 3b + c = 3 + d \end{cases}.$$

- D'où

$$\begin{cases} a + b + c = 0 \\ 4a + 2b + c - 3d = 1 \\ a + 4b + c = 0 \\ 4a + 3b + c - d = 3 \end{cases}.$$

- Ce qui nous permet d'obtenir après résolution $a = 3$, $b = 0$, $c = 2$ et $d = 1$.
- Donc $N(x) = 3x^2 + 2 = f(x)(x + 1)$. Par une division Euclidienne, on a $f(x) = 3x + 2$.

Exemple

$\mathbf{a} = (1, 2, 4, 3)$.

Nous allons décoder $\mathbf{y} = (0, 3, 0, 1)$. On a :

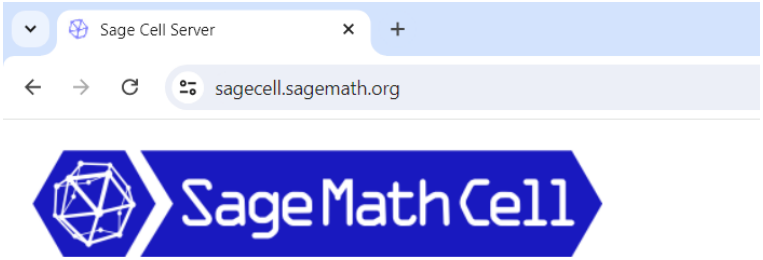
- $\forall i \in [|1; 4|] \quad y_i \times \mathcal{L}_e(a_i) = N(a_i)$.
- De plus $deg(N) = 2$ et $deg(\mathcal{L}_e) = 1$; d'où $N(x) = ax^2 + bx + c$ et $\mathcal{L}_e(x) = x + d$.
- On a alors le système suivant :

$$\begin{cases} a + b + c = 0 \\ 4a + 2b + c = 3(2 + d) \\ a + 4b + c = 0 \\ 4a + 3b + c = 3 + d \end{cases}.$$

- D'où

$$\begin{cases} a + b + c = 0 \\ 4a + 2b + c - 3d = 1 \\ a + 4b + c = 0 \\ 4a + 3b + c - d = 3 \end{cases}.$$

- Ce qui nous permet d'obtenir après résolution $a = 3$, $b = 0$, $c = 2$ et $d = 1$.
- Donc $N(x) = 3x^2 + 2 = f(x)(x + 1)$. Par une division Euclidienne, on a $f(x) = 3x + 2$.


Sage Cell Server
sagecell.sagemath.org
SageMathCell
Type some Sage code below and press Evaluate.
# On peut utiliser SageMath pour resoudre les systemes lineairs AX=Y.
A = Matrix(GF(5), [[1,1,1,0],[4,2,1,-3],[1,4,1,0],[4,3,1,-1]])
Y = vector([0,1,0,3])
X = A.solve_right(Y)
print('X=',X)
Evaluate
X= (3, 0, 2, 1)

```sage
# Les codes de Reed Solomon sont implementes dans SageMath .
F = GF(5)
Fx.<x> = F[] # Espace des polynomes
n, k =4, 2  # Paremetres du codes
a = [F(1),F(2),F(4),F(3)] # Support
C = codes.GeneralizedReedSolomonCode(a, k) # Code de Reed Salomon.
E = C.encoder("EvaluationPolynomial")
G=C.generator_matrix() # Matrice generatrice
p =  2+3*x # Polynome representatif du message m=(2,3).
c = E.encode(p)  # Encodeage du message.
e=vector(F, [0, 0, 1, 0]) # L'erreur
y= c+e # Mot recu
D = codes.decoders.GRSBerlekampWelchDecoder(C) # Algorithme de decodage
v=D.decode_to_code(y) # Decodage du mot recu.
print('Support: a=',a)
print('Matrice generatice: G=')
print(G)
print('Encodeage du message: c=',c)
print('Mot recu: y=',y)
print('Decodage du mot recu: v=',v)
```

Evaluate

Language: Sage

Share

```
Support: a= [1, 2, 4, 3]
Matrice generatice: G=
[1 1 1 1]
[1 2 4 3]
Encodeage du message: c= (0, 3, 4, 1)
Mot recu: y= (0, 3, 0, 1)
Decodage du mot recu: v= (0, 3, 4, 1)
```

Help | Powered by SageMath

Codes de Reed-Solomon Généralisés

- $\mathbf{a} = (a_1, ..., a_n)$, où les a_i sont des éléments distincts de $\mathbb{F}_q$
- $\mathbf{b} = (b_1, ..., b_n)$ où les b_i sont des éléments non-nuls

Définition 22

Le code de Reed-Solomon Généralisé $GRS_k(\mathbf{a}, \mathbf{b})$ sur $\mathbb{F}_q$ est l'ensemble :

$$GRS_k(\mathbf{a}, \mathbf{b}) = \{((b_1 f(a_1), ..., \mathbf{b}_n f(a_n)) \ / \ f \in \mathbb{F}_q[X] \ , deg(f) < k\}$$

Codes de Reed-Solomon Généralisés

Proposition 23

.

(i) *$GRS_k(\boldsymbol{a}, \boldsymbol{b})$ est un code linéaire de longueur n, de dimension k.*

(ii) *Une matrice génératrice de $GRS_k(\boldsymbol{a}, \boldsymbol{b})$ est*

$$\boldsymbol{G} = \begin{pmatrix} b_1 & b_2 & . & . & . & . & b_n \\ b_1 a_1 & b_2 a_2 & . & . & . & . & b_n a_n \\ b_1 a_1^2 & b_2 a_2^2 & . & . & . & . & b_n a_n^2 \\ . & . & & & & & . \\ . & . & & & & & . \\ . & . & & & & & . \\ b_1 a_1^{k-1} & b_2 a_2^{k-1} & . & . & . & . & b_n a_n^{k-1} \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 1 & . & . & . & . & 1 \\ a_1 & a_2 & . & . & . & . & a_n \\ a_1^2 & a_2^2 & . & . & . & . & a_n^2 \\ . & . & & & & & . \\ . & . & & & & & . \\ . & . & & & & & . \\ a_1^{k-1} & a_2^{k-1} & . & . & . & . & a_n^{k-1} \end{pmatrix} \begin{pmatrix} b_1 & & & 0 \\ & b_2 & & \\ & & & \\ & & & \\ & & & \\ 0 & & & b_n \end{pmatrix}$$

Proposition 24

(iii) *$GRS_k(\boldsymbol{a}, \boldsymbol{b})$ est MDS.*

(iv) *L'orthogonal de $GRS_k(\boldsymbol{a}, \boldsymbol{b})$ est $GRS_{n-k}(\boldsymbol{a}, \boldsymbol{w})$, c'est-à-dire, $GRS_k(\boldsymbol{a}, \boldsymbol{b})^{\perp} = GRS_{n-k}(\boldsymbol{a}, \boldsymbol{w})$, où les composantes de $\mathbf{w} = (w_1, ..., w_n)$ sont :*

$$w_i = \frac{1}{b_i F'(a_i)} \quad (1 \leqslant i \leqslant n),$$

avec $F(x) = \prod_{i=1}^{n} (x - a_i)$.

Remark 25

Le principe du décodage des codes de Reed-Solomon généralisés est le même que celui des codes de Reed-Solomon. En effet, si le mot reçut est $\mathbf{y} = \mathbf{m}\boldsymbol{G}' + \mathbf{e}$ (avec $\boldsymbol{G}' = \boldsymbol{G}\boldsymbol{S}$; où $\boldsymbol{G}$ est une matrice génératrice de $RS_k(\boldsymbol{a})$ et $\boldsymbol{S}$ une matrice diagonale inversible $n \times n$. On calcule alors $\mathbf{y}\boldsymbol{S}^{-1} = (\mathbf{m}\boldsymbol{G}' + \mathbf{e})\boldsymbol{S}^{-1} = \mathbf{m}\boldsymbol{G} + \mathbf{e}\boldsymbol{S}^{-1}$. Retrouver $\mathbf{m}$ revient alors à décoder un mot du code de Reed-Solomon $RS_k(\boldsymbol{a})$.

```sage
# Les codes de Reed Solomon generalise sont implementes dans SageMath .
F = GF(5)
Fx.<x> = F[] # Espace des polynomes
n, k =4, 2  # Paremetres du codes
a = [F(1),F(2),F(4),F(3)] # Support pour evaluation
b=[F(1),F(1),F(2),F(3)] # Support pour multiplication
C = codes.GeneralizedReedSolomonCode(a,k,b) # Code de Reed Salomon generalise.
E = C.encoder("EvaluationPolynomial")
G=C.generator_matrix() # Matrice generatrice
Cd = C.dual_code() # Code dual du code C
H=Cd.generator_matrix() # Matrice de controle
v=Cd.evaluation_points() # Support pour evaluation de dual
w=Cd.column_multipliers() # Support pour multiplication de dual
p =  2+3*x # Polynome representatif du message m=(2,3).
c = E.encode(p)  # Encodeage du message.
e=vector(F, [0, 0, 1, 0]) # L'erreur
y= c+e # Mot recu
D = codes.decoders.GRSBerlekampWelchDecoder(C) # Algorithme de decodage
z=D.decode_to_code(y) # Decodage du mot recu.
print('Matrice generatice: G=')
print(G)
print('Matrice controle: H=')
print(H)
print('Support pour evaluation du dual : v=',v)
print('Support pour multiplication du dual w=',w)
print('Encodeage du message: c=',c)
print('Mot recu: y=',y)
print('Decodage du mot recu: z=',z)
```

```
Matrice generatice: G=
[1 1 2 3]
[1 2 3 4]
Matrice controle: H=
[4 3 3 4]
[4 1 2 2]
Support pour evaluation du dual : v= (1, 2, 4, 3)
Support pour multiplication du dual w= (4, 3, 3, 4)
Encodeage du message: c= (0, 3, 3, 3)
Mot recu: y= (0, 3, 4, 3)
Decodage du mot recu: z= (0, 3, 3, 3)
```


Codes parfaits
Codes MDS
Codes MDS parfaits
En blocs
Linéaire
Non linéaire
Constacyclique
Quasicyclique
Polynomial
Géométrique
Cyclique
Goppa géométrique
Reed-Muller
Torique
Golay
Duadique
Elliptique
Reed-Solomon généralisé
Hadamard
Résidu Quadratique
Cauchy
Alternant
Goppa classique
BCH généralisé
Srivastava généralisé
Hamming
BCH
Srivastava
Gabidulin
Tetracode
Simplexe
BCH primitif
BCH au sens strict
Reed-Solomon
BCH primitif au sens strict
Reed-Solomon étendu
Justesen
Reed-Muller cyclique

Plan

Exercice 1 : Distance Minimale et matrice de contrôle

1. Soit C un code linéaire avec une matrice de contrôle H et le distance minimale d.
   1. Montrer que si r colonnes de H sont linéairement dépendantes, alors $d \leqslant r$.
   2. Montrer que si toutes les r colonnes de H sont linéairement indépendantes, alors $r < d$.
2. Soit C un code linéaire sur $\mathbb{F}_5$ de matrice de contrôle :

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \end{pmatrix}$$

   1. Déterminer la distance minimale de C.
   2. Déterminer une matrice génératrice de C.
   3. Un mot reçu du code C est $y = \begin{pmatrix} 2 & 1 & 2 & 1 \end{pmatrix}$.
      En supposant que le nombre d'erreurs survenues au cours la transmission est inférieure ou égale à la capacité de correction d'erreur. Trouver le mot de code transmis et le message transmis.

Exercise 2 (Cyclic Codes)

Let C be a linear code over $\mathbb{F}_q$ of length n.

C is a cyclic code if for every $(c_1, \ldots, c_{n-1}, c_n)$ in C, the word $(c_n, c_1, \ldots, c_{n-1})$ is in C.

Let $(X^n - 1)$ be the ideal of $\mathbb{F}_q[X]$ generated by $X^n - 1$ and $\mathbb{F}_q[X]/(X^n - 1)$ the quotient ring.

Let the map $\Psi : \mathbb{F}_q^n \longrightarrow \mathbb{F}_q[X]/(X^n - 1)$ defined by
$\Psi((c_1, c_2, \ldots, c_n)) = c_1 + c_2X + \cdots + c_nX^{n-1}$.

If $c = (c_1, c_2, \ldots, c_n)$ is in $\mathbb{F}_q^n$, then $\Psi((c_1, c_2, \ldots, c_n))$ is denoted by $c(X)$, that is,
$c(X) = c_1 + c_2X + \cdots + c_nX^{n-1}$.

1. Prove that C is a cyclic code if and only if $\Psi(C)$ is an ideal of $\mathbb{F}_q[X]/(X^n - 1)$.
2. Prove that if C is a cyclic code then the ideal $\Psi(C)$ is generated by the unique monic polynomial $g(X)$. This polynomial is called the generator polynomial of C.
3. Give the dimension and a generator matrix of a cyclic code.
4. Show that if C is a cyclic code then the dual $C^\perp$ is also a cyclic code and specify the generator polynomial of $C^\perp$.
5. Let $C = \{(x_1, x_2, -x_1 - x_2),\ x_1 \in \mathbb{F}_q, x_2 \in \mathbb{F}_q\}$.
   1. Show that C is a cyclic code.
   2. Give a generator matrix and the generator polynomial of C and $C^\perp$.