Identification aveugle de codes correcteurs d'erreurs basés sur des grands corps de Galois et recherche d'algorithmes de type décision souple pour les codes convolutifs

Yasamine Zrelli

université de bretagne occidentale

THÈSE / UNIVERSITÉ DE BRETAGNE OCCIDENTALE
sous le sceau de l'Université Européenne de Bretagne

pour obtenir le titre de
DOCTEUR DE L'UNIVERSITÉ DE BRETAGNE OCCIDENTALE
Mention : Sciences et Technologies de l'Information et de la Communication - Spécialité Communications Numériques
École Doctorale SICMA-ED 373

présentée par
Yasamine ZRELLI
Préparée au Lab-STICC UMR-CNRS 6285

Identification aveugle de codes correcteurs d'erreurs basés sur des grands corps de Galois et recherche d'algorithmes de type décision souple pour les codes convolutifs

Thèse soutenue le 10 décembre 2013
devant le jury composé de :

Maryline HELARD
Professeur des Universités, IETR, INSA de Rennes / *Examinatrice*

Pascal CHARGE
Professeur des Universités, IETR, Polytech Nantes / *Rapporteur*

Pierre LOIDREAU
Expert à la DGA, chercheur à l'IRMAR, Université de Rennes 1 / *Rapporteur*

Roland GAUTIER
Maître de Conférences, UBO, Lab-STICC / *Examinateur*

Eric RANNOU
Maître de Conférences, UBO, Laboratoire de Mathématiques / *Examinateur*

Emanuel RADOI
Professeur des Universités, UBO, Lab-STICC / *Directeur de thèse*

Sébastien HOUCKE
Maître de Conférences, Télécom-Bretagne / *Invité*

Mélanie MARAZIN
Maître de Conférences, UBO, Lab-STICC / *Invité*

Table des matières

Liste des acronymes et abréviations

APP	A Posteriori Probability
BBAG	Bruit Blanc Additif Gaussien
BPSK	Binary Phase-Shift Keying
BSC	Binary Symmetric Channel
dB	Décibel
GF	Galois Field
GSM	Global System for Mobile Communications
LDPC	Low Density Parity Check
LRV	Logarithme du Rapport de Vraisemblance
LTE	Long Term Evolution
MAP	Maximum A posteriori
MMSE	Minimum Mean Square Error
NRNSC	Non-Recursive and Non-Systematic Code
PAM	Pulse Amplitude Modulation
PGCD	Plus Grand Commun Diviseur
QAM	Quadrature Amplitude Modulation
QoS	Quality of Service
QSC	Q-ary Symmetric Channel
RSB	Rapport Signal à Bruit
RSC	Recursive and Systematic Code
TEB	Taux d'Erreur Binaire
TES	Taux d'Erreur Symbole
UMTS	Universal Mobile Telecommunications System

Notations

Pour toute matrice comprenant de nombreux “0”, comme la matrice $\mathbf{A}$ dans l'exemple ci-dessous, des “0” pourront être omis afin de simplifier la notation.

$$\mathbf{A} = \begin{pmatrix} a_1 & 0 & 0 & 0 \\ 0 & a_2 & 0 & 0 \\ 0 & 0 & a_3 & 0 \\ 0 & 0 & 0 & a_4 \end{pmatrix} = \begin{pmatrix} a_1 & & & \\ & a_2 & & \\ & & a_3 & \\ & & & a_4 \end{pmatrix}$$

Notations mathématiques

$\forall$	Quel que soit
$\exists$	Il existe
$\propto$	Proportionnel à
$\wedge$	Le plus grand commun diviseur
$\vee$	Le plus petit commun multiple
$\lceil x \rceil$	Arrondi de x à l'entier supérieur
$\mathrm{mod}(a, b)$	a modulo b
$Pr\,[\cdot]$	Probabilité
σ	L'endomorphisme de Frobenius
$\mathrm{GF}(q)$=$\mathbb{F}_q$	Corps de Galois de cardinal q
$\mathrm{GF}(q)[x]$=$\mathbb{F}_q[x]$	Anneau des polynômes dont les coefficients appartiennent à $\mathrm{GF}(q)$= $\mathbb{F}_q$
$\mathbb{K}\backslash\{0\} = \mathbb{K}^*$	Groupe des éléments inversibles de $\mathbb{K}$, c'est-à-dire non nuls
$\mathbb{Z}/p\mathbb{Z}$	Le plus petit corps de caractéristique p
$p(x)$	Polynôme primitif
$\mathbf{A}$	Matrice composée d'éléments dans $\mathrm{GF}(q)$
$a_{i,j}$	Le coefficient de la matrice $\mathbf{A}$ situé à la i-ème ligne et à la j-ème colonne
$\ker(\mathbf{A})$	Noyau de la matrice $\mathbf{A}$
$\mathbf{A} = (\cdot)$	$\mathbf{A}$ est une matrice vide
$(.)^T$	Vecteur ou matrice transposé(e)
$\mathbf{I}_N$	Matrice identité de taille $(N \times N)$

Notations mathématiques des codes correcteurs d'erreurs

n	Taille des mots de code
k	Taille des mots d'information

K	Longueur de contrainte
R	Rendement du code
μ_i	Mémoire de la i-ème entrée du codeur
μ	Mémoire du codeur
$\mu^{\perp}$	Mémoire du code dual
$\mathcal{C}(n,k)$	Code en bloc de paramètres n et k
LDPC(n,k)	Code LDPC de paramètres n et k
RS(n,k)	Code de Reed-Solomon de paramètres n et k
$\mathcal{C}(n,k,K)$	Code convolutif de paramètres n, k et K
$\mathbf{u}$	Vecteur d'entrée du codeur
$u(x)$	Représentation polynomiale d'un mot d'information
$\mathbf{c}$	Vecteur de sortie du codeur
$c(x)$	Représentation polynomiale d'un mot de code
$\mathbf{e}$	Vecteur d'erreurs introduites par le canal
$\mathbf{r}$	Vecteur de données reçues à la sortie du canal
$\mathbf{G}$	Matrice génératrice
$g(x)$	Polynôme générateur d'un code cyclique
$\mathbf{g}_{i,j}$	Vecteur composé des coefficients du (i,j)-ème polynôme générateur
$g_{i,j}(D)$	Transformée en D du polynôme générateur $\mathbf{g}_{i,j}$
$g_{i,j}(l)$	Le l-ième coefficient du (i,j)-ème polynôme générateur $(\mathbf{g}_{i,j})$
$\mathbf{F}$	Matrice de codage composée des sous-matrices de codage
$\mathbf{F}_l$	La l-ième sous-matrice de codage
$\mathcal{C}^{\perp}$	Code dual
$\mathbf{H}$	Matrice de contrôle de parité
$\mathbf{H}_l$	La l-ième sous-matrice de contrôle de parité
$d_H(\mathbf{c},\mathbf{c}')$	Distance de Hamming entre les deux vecteurs $\mathbf{c}$ et $\mathbf{c}'$
$w_H(\mathbf{c})$	Poids de Hamming du vecteur $\mathbf{c}$
d_{min}	Distance minimale du code
p_e	Probabilité d'erreur du canal
$\deg g(D)$	Degré du polynôme $g(D)$

Notations spécifiques du chapitre 3

L	Taille des données reçues
n_a	Taille de la première matrice de rang déficient
$\mathbf{R}_l$	Matrice de taille $(M \times l)$ composée des éléments de $\mathbf{c}$
$\mathbf{R}_{l,i}$	La i-ème sous-matrice de $\mathbf{R}_l$
$rang(\mathbf{R}_l)$	Rang de la matrice $\mathbf{R}_l$
$\mathbf{T}_n$	Matrice obtenue par la transformation de $\mathbf{R}_n$ par l'élimination de Gauss $(\mathbf{T}_n = \mathbf{A}_n \cdot \mathbf{R}_n)$
$\mathbf{t}_i^{(n)}$	La i-ème colonne de la matrice $\mathbf{T}_n$
$N_n(i)$	Nombre des zéros dans la colonne $\mathbf{t}_i^{(n)}$
$\mathcal{D}$	Base du code dual
$\mathbf{h}_i$	La i-ème relation de parité du code

Notations spécifiques du chapitre 4

$\tilde{\mathbf{R}}_l$	Matrice de taille $(M \times l)$ composée des éléments de $\mathbf{r}$
$Q(l)$	Nombre de colonnes presque dépendantes dans la matrice $\tilde{\mathbf{R}}_l$
k_l	Nombre de colonnes indépendantes dans la matrice $\tilde{\mathbf{R}}_l$
$\tilde{\mathbf{T}}_l$	Matrice obtenue par la transformation de $\tilde{\mathbf{R}}_l$ par l'élimination de Gauss
$B_l(i)$	Nombre de zéros dans la colonne $\tilde{\mathbf{t}}_i^{(l)}$
$N_i(l)$	Nombre minimal de combinaisons linéaires requises pour obtenir la colonne $\tilde{\mathbf{t}}_i^{(l)}$
P_i	Probabilité d'avoir un coefficient nul dans la colonne $\tilde{\mathbf{t}}_i^{(l)}$
E_l	Moyenne arithmétique des variables $B_l(i)$
V_l	Variance empirique des variables $B_l(i)$
$\mathbf{E}_l$	Matrice de taille $(M \times l)$ composée d'erreurs introduites par le canal
γ	Seuil de détection (pour les codes binaires)
η	Seuil de détection (pour les codes non-binaires)
$\tilde{n}$	Taille identifiée de mots de code
$c_i^{(t)}$	Le i-ème élément du t-ème mot de code
L_c	Fiabilité du canal
L_{path}	Nombre de trajets dans un canal de Rayleigh multi-trajets
$L(\mathbf{r})$	Logarithme du rapport de vraisemblance de $\mathbf{r}$
LRV_{ext}	Logarithme du rapport de vraisemblance extrinsèque
$\hat{\mathbf{R}}_l$	Matrice de taille $(M \times l)$ composée des informations souples
$F(j)$	Estimateur de la fiabilité de la j-ème ligne de $\hat{\mathbf{R}}_l$
$Z(j)$	Estimateur du nombre de bits fiables dans la j-ème ligne de $\hat{\mathbf{R}}_l$
S	Seuil de fiabilité de la décision souple
z_i	La i-ème équation de parité (dans le cadre de l'algorithme de décodage)
p_i	Probabilité de fiabilité de l'équation de parité z_i
$\tilde{p}_i(t)$	Probabilité de fiabilité de l'équation de parité z_i par le t-ème mot de code
$\tilde{c}$	Mot de code corrigé par le décodeur
nb_{Eq}	Nombre d'équations de parité identifiées
$\mathbf{D}$	Matrice de taille $(nb_{Eq} \times n)$ composée des équations de parité identifiées
$\mathbf{p}$	Matrice composée des probabilités de fiabilité des équations de parité estimées
$\tilde{\mathbf{C}}$	Matrice composée des mots de code corrigés
$iter$	Nombre d'itérations maximal de l'algorithme de décodage
it_{max}	Nombre d'itérations maximal du processus d'identification
$\lambda_{x \to y}$	Gain obtenu entre l'itération x et l'itération y
P_{det}	Probabilité de détection
P_{fa}	Probabilité de fausse alarme
P_{wd}	Probabilité de non détection

Notations spécifiques des chapitres 5 et 6

id	Opérateur identité
$\rtimes$	Produit semi-direct
$\circ$	Loi de composition de deux applications

$\dim(V)$	Dimension d'un espace vectoriel V
τ_μ	Translation par un scalaire μ de $\mathrm{GF}(p^m)^*$
$\varphi(t)$	Fonction d'Euler
$(\mathbf{1})_i$	Vecteur composé uniquement de 1
m_λ	Application de multiplication par un scalaire $\lambda \in \mathrm{GF}(p^m)^*$
d^{-1}	Fonction réciproque de la fonction d
$(m_{i,j})_{i,j}$	Matrice $\mathbf{M}$ ayant i l'indice des lignes et j l'indice des colonnes
$(x_i)_{1\leq i\leq n}$	Vecteur $\mathbf{x}$ de longueur n
$\overline{\mathbb{MD}}$	Groupe des fonctions de mapping-demapping universelles
g	Fonction de mapping-demapping universelle
$\lvert\overline{\mathbb{MD}}\rvert$	Cardinal du groupe $\overline{\mathbb{MD}}$
$\widetilde{\mathbb{MD}}$	Groupe des fonctions de mapping-demapping quasi-universelles
h	Fonction de mapping-demapping quasi-universelle
$\mathbb{MD}$	Groupe des toutes les fonctions de mapping-demapping
N_t	Nombre de fonctions de mapping-demapping conservant une relation de parité de poids 2

Liste des figures

Liste des Algorithmes

Liste des tableaux

Introduction

Contexte et objectif

Pour obtenir un système de communication fiable, il est indispensable d'intégrer dans sa chaîne de traitement un bloc de codage canal comprenant au moins un code correcteur d'erreurs. En effet, les codes correcteurs d'erreurs ont pour rôle de protéger les bits ou les symboles informatifs par l'ajout de bits ou de symboles redondants qui permettent, coté récepteur, de détecter et/ou corriger d'éventuelles erreurs. A cause de la complexité de l'encodage et en particulier des procédures du décodage, la majorité des recherches et des implémentations pratiques dans des systèmes embarqués en temps réel ont été souvent limitées à des codes manipulant des données binaires, c'est-à-dire traitant des éléments du corps de Galois GF(2).

Dans le cadre de cette thèse, nous nous sommes intéressés au cas des codes correcteurs d'erreurs construits sur des corps de Galois non-binaires de cardinal 2^m. Parmi les codes dans GF(2^m), il existe les codes LDPC non-binaires qui ont montré récemment de meilleurs performances que les codes LDPC binaires. Dans nos travaux, nous nous sommes intéressés à ces codes, mais également aux codes, peu classiques, de Reed-Solomon et aux codes convolutifs non-binaires. Dans la première partie de ce mémoire, notre objectif principal est d'identifier en aveugle les paramètres de ces codes.

Le thématique de l'identification aveugle en contexte non-coopératif concerne l'interception militaire ou la surveillance du spectre, mais également les applications de la radio intelligente. Dans ce contexte, le récepteur ne connaît pas les paramètres utilisés à l'émission. La solution est de concevoir un récepteur intelligent qui sera capable d'identifier en aveugle les paramètres de l'émetteur à partir de la seule connaissance des données reçues. Un des intérêts de l'identification aveugle est d'augmenter le débit utile de transmission, car il ne sera plus nécessaire de transmettre des informations supplémentaires sur les paramètres du codeur avec les données utiles. De plus, un tel récepteur intelligent sera capable de s'adapter automatiquement au développement de nouveaux schémas de codage plus performants et à l'évolution rapide des nouvelles normes et standards de communications.

Dans un contexte non-coopératif, le récepteur ne connaît généralement pas le mapping utilisé à l'émission. Pour cela, il doit identifier l'opération inverse du mapping, appelée demapping, qui permet de récupérer les symboles convertis par le mapping avant la transmission. En effet, l'opération de mapping dans le cas des codes non-binaires consiste à associer à un élément du corps de Galois une représentation spécifique. Pour récupérer les bons symboles transmis, la composition du mapping et du demapping, appelée fonction de mapping-demapping, doit être égale à l'opérateur identité. Dans certains cas, cette composition peut ne pas être égale à l'opérateur identité. Ce phénomène peut se produire à cause d'une mauvaise identification de l'opération appropriée de demapping dans un contexte non-coopératif. Les problèmes de synchronisation peuvent également

produire involontairement ce phénomène.

Dans la deuxième partie de ce mémoire, notre objectif est donc d'étudier l'influence des différentes fonctions de mapping-demapping sur les calculs d'algèbre linéaire et sur les propriétés dans le corps de Galois utilisé dans une chaîne de transmission pour des communications numériques. Pour cela, un formalisme mathématique est proposé. En utilisant ce formalisme, nous allons étudier l'impact des fonctions de mapping-demapping sur la détection de certains défauts de transmission et sur l'identification aveugle des paramètres des codes correcteurs d'erreurs non-binaires.

Plan du mémoire

Comme nous l'avons indiqué précédemment, ce document est organisé en deux parties. La première partie qui porte sur l'identification aveugle des codes correcteurs d'erreurs non-binaires est constituée de quatre chapitres :

Le chapitre 1

Ce chapitre présente un rappel sur des différentes structures algébriques afin d'introduire la structure des corps de Galois. Notre objectif est de générer les symboles du corps de Galois qui vont être traités par les codes correcteurs d'erreurs non-binaires. Cette étude sur les corps de Galois nous permettra de connaître les paramètres indispensables pour générer leurs éléments.

Le chapitre 2

Ce chapitre a pour objet d'introduire les blocs de traitement dans une chaîne de transmission numérique, ainsi que les différents modèles de canaux de transmission qui vont être considérés dans cette thèse. Ensuite, nous présentons les propriétés des codes linéaires dans $\mathrm{GF}(q)$, ainsi que les codes abordés dans nos travaux afin de faire ressortir les propriétés intéressantes et les paramètres de codage utiles pour les identifier au niveau du récepteur.

Le chapitre 3

Ce chapitre s'intéresse à l'identification des codes correcteurs d'erreurs non-binaires en supposant une transmission non-bruitée. Une généralisation de l'algorithme d'identification pour les codes binaires basée sur le critère du rang est présentée. Ensuite, une justification théorique de l'utilisation de ce critère est développée. L'identification des codes aux paramètres spécifiques est également étudiée. Enfin, nous présentons une étude de l'impact des paramètres du corps de Galois sur l'identification aveugle des paramètres des codes correcteurs d'erreurs non-binaires.

Le chapitre 4

Dans ce chapitre, une transmission entachée d'erreurs est considérée. Nous développons tout d'abord trois algorithmes d'identification de la taille des mots de code pour des codes non-binaires et binaires. Ensuite, nous analysons les performances de ces méthodes. Dans le but d'identifier en aveugle une base du code dual, nous généralisons une méthode d'identification à décision ferme pour les codes binaires au cas des codes non-binaires. Afin d'améliorer les performances de détection de l'algorithme d'identification, une méthode basée sur l'utilisation conjointe d'un démodulateur à décision souple et d'un décodage itératif à décision souple est présentée. Nous introduisons un processus itératif basé sur l'échange des informations souples et des relations de parités identifiées entre

l'algorithme d'identification et l'algorithme de décodage à décision souple. Enfin, nous analysons les performances des méthodes présentées.

Dans la seconde partie de ce document, nous nous intéressons aux fonctions de mapping-demapping. Cette partie comporte deux chapitres :

Le chapitre 5

Dans ce chapitre, nous développons un formalisme mathématiques afin d'étudier l'impact des fonctions de mapping-demapping sur les calculs d'algèbre linéaire dans un corps de Galois effectués dans les blocs d'une chaîne de transmission pour des communications numériques. Deux fonctions de mapping-demapping particulières correspondant au changement de polynôme primitif et à une inversion des poids sont également étudiées.

Le chapitre 6

Dans ce chapitre, nous étudions tout d'abord l'impact des fonctions de mapping-demapping sur la détection des défauts de transmission rencontrés au niveau du récepteur et sur la détection des défauts liés à une mauvaise interprétation des éléments des corps de Galois. Ensuite, nous étudions l'impact de certaines fonctions de mapping-demapping sur l'identification des paramètres des codes correcteurs d'erreurs dans un contexte de transmission non-bruitée.

Enfin, ce document se conclut en rappelant les différentes étapes et résultats de cette thèse et en proposant plusieurs pistes de recherche et perspectives de ce travail.

Première partie

Identification aveugle des codes correcteurs d'erreurs

CHAPITRE 1

Rappel algébrique sur les corps de Galois

1.1 Introduction

Ce premier chapitre a pour but de montrer comment représenter un symbole non-binaire d'un corps de Galois de cardinal 2^m, $m > 1$. De ce fait, un rappel sur les notions des corps finis est indispensable. Dans un premier temps, nous allons faire un rappel sur les différentes structures algébriques afin d'introduire les structures des corps finis, c'est-à-dire celles des corps de Galois. La dernière partie de ce chapitre exposera la méthode de construction des corps de Galois $\mathrm{GF}(2^m)$. Cette méthode sera illustrée par la construction du corps $\mathrm{GF}(2^4)$.

1.2 Structures algébriques

Les structures algébriques sont très utilisées en codage et en cryptographie. Une structure algébrique comporte deux types de constituants : les objets et les morphismes. Les objets sont le plus souvent des ensembles munis d'une ou plusieurs opérations qui vérifient certaines propriétés. Le choix des propriétés imposées caractérise la structure algébrique. Les morphismes sont très généralement des applications entre deux objets de la structure qui sont compatibles avec les lois et les propriétés typiques de la structure. Le tableau 1.1 présente des exemples de structures algébriques et les morphismes qui leur correspondent.

Structures algébriques	**Objets**	**Morphismes**
Catégorie des ensembles	Ensembles	Applications
Catégorie des groupes	Groupes	Morphismes de groupes
Catégorie des corps	Corps	Morphismes de corps
Catégorie des espaces vectoriels sur $\mathbb{R}$	Espaces vectoriels sur $\mathbb{R}$	Applications linéaires

Tableau 1.1 — Les structures algébriques usuelles

L'ensemble des entiers relatifs $\mathbb{Z}$ muni de l'opération d'addition est un exemple d'un objet de la catégorie groupe.

1.2.1 Groupes

Définition 1.1. *Un groupe est un ensemble G muni d'une opération notée $*$. Un ensemble $(G, *)$ est un groupe si les propriétés suivantes sont vérifiées :*

1. *G est non vide,*

2. *$*$ est associative,*

3. *G admet un élément neutre e à gauche et à droite : $\forall x \in G$, $x * e = x = e * x$,*

4. *tout élément de G admet un inverse pour $*$: $\forall x \in G$, $\exists x^{-1} \in G$ tel que $x * x^{-1} = e = x^{-1} * x$.*

Si la loi $*$ est commutative (i.e $\forall (x, y) \in G^2, x * y = y * x$), on dit que $(G, *)$ est commutatif, ou encore abélien.

Exemple 1.1.

- $(\mathbb{Z}, +)$, $(\mathbb{R}, +)$ et $(\mathbb{C} \setminus \{0\}, \cdot)$ sont des groupes abéliens.
- Le groupe des matrices de taille (2×2) à coefficients réels et munis de l'opération d'addition, noté $(\mathbb{M}_2(\mathbb{R}), +)$, est également un groupe abélien.
- Le groupe des matrices inversibles de taille (2×2) à coefficients réels et munis de l'opération de multiplication, noté $(GL_2(\mathbb{R}), \cdot)$, n'est pas abélien.

1.2.2 Anneaux

Définition 1.2. *Un groupe commutatif A est un anneau, noté $(A, +, \cdot)$, s'il est muni d'une seconde loi de composition interne vérifiant les propriétés suivantes :*

- *$(A, +)$ est un groupe abélien d'élément neutre 0_A,*
- *la loi $\cdot$ est associative : $\forall (x, y, z) \in G^3$, $(x \cdot y) \cdot z = x \cdot (y \cdot z)$,*
- *la loi $\cdot$ est distributive à gauche et à droite par rapport à la loi $+$: $\forall (x, y, z) \in G^3$, $x \cdot (y + z) = x \cdot y + x \cdot z$ et $(x + y) \cdot z = x \cdot z + y \cdot z$,*
- *la loi $\cdot$ admet un élément neutre différent de 0_A, noté 1_A.*

Exemple 1.2.

$(\mathbb{R}, +, \cdot)$ est un anneau commutatif, car la loi $\cdot$ est aussi commutative.

Les anneaux et les corps sont des structures algébriques plus complexes que les groupes car ils sont munis de plus d'une opération. Les corps sont les structures qui nous intéressent dans le cadre de notre étude.

1.2.3 Corps

Définition 1.3. *Un corps $\mathbb{K}$ est un anneau commutatif dans lequel tout élément non-nul est inversible. Soit 0 l'élément neutre pour l'addition et 1 celui de la multiplication, pour tout $\alpha \in \mathbb{K}$ il doit exister $-\alpha$ et α^{-1} tel que :*

- *$0 + \alpha = \alpha$ et $(-\alpha) + \alpha = 0$,*
- *$1 \cdot \alpha = \alpha$ et $0 \cdot \alpha = 0$,*
- *si $\alpha \neq 0$ alors $(\alpha^{-1}) \cdot \alpha = 1$.*

Exemple 1.3.

L'ensemble des entiers relatifs $(\mathbb{Z}, +, \cdot)$ est un anneau unitaire mais pas un corps car l'élément 2, par exemple, n'a pas d'inverse.
Les ensembles $\mathbb{Q} = \left\{ \frac{p}{q}, p \in \mathbb{Z}, q \in \mathbb{Z}, q \neq 0 \right\}$, $\mathbb{R}$ et $\mathbb{C}$ sont des corps.

1.2.4 Espace vectoriel sur un corps

Un espace vectoriel sur $\mathbb{R}$ est défini par un ensemble V dont les éléments sont appelés vecteurs, muni de deux opérations : l'opération d'addition de deux vecteurs et l'opération de multiplication d'un vecteur par un nombre réel. Dans ce cas, l'addition est une loi interne et la multiplication par un scalaire est une loi externe.

Définition 1.4. *Un espace vectoriel sur le corps $\mathbb{K}$ est un ensemble E muni de deux lois de composition :*

- *Une loi interne notée $+$, telle que $(E, +)$ est un groupe abélien, c'est-à-dire une application $E \times E \to E$, qui vérifie les propriétés suivantes :*
  - $\forall (u, v) \in E^2$, $u + v = v + u$,
  - $\forall (u, v, w) \in E^3$, $(u + v) + w = u + (v + w)$,
  - *il existe un élément neutre de E noté 0,*
  - *pour tout élément u, il existe un élément symétrique noté $(-u)$ appelé opposé.*
- *Une loi externe, c'est-à-dire une application de $\mathbb{K} \times E \to E$, notée $\cdot$, qui vérifie les propriétés suivantes, $\forall (a, b) \in \mathbb{K}^2$, $\forall (u, v) \in E^2$:*
  - $a \cdot (u + v) = a \cdot u + a \cdot v$,
  - $(a + b) \cdot u = a \cdot u + b \cdot u$,
  - $a \cdot (b \cdot u) = (a \cdot b) \cdot u$,
  - $1 \cdot u = u$.

Exemple 1.4.

- $\mathbb{R}$ est un espace vectoriel sur $\mathbb{Q}$.
- Soit $\mathbb{K}$ un corps, l'ensemble des matrices à p lignes et q colonnes, noté $\mathbb{M}_{p,q}(\mathbb{K})$, est un espace vectoriel sur le corps $\mathbb{K}$. En effet, si $\mathbf{A} = (a_{i,j})_{i \in [\![1,p]\!],\, j \in [\![1,q]\!]} \in \mathbb{M}_{p,q}(\mathbb{K})$ et $\mathbf{B} = (b_{i,j})_{i \in [\![1,p]\!],\, j \in [\![1,q]\!]} \in \mathbb{M}_{p,q}(\mathbb{K})$, les lois de l'ensemble $\mathbb{M}_{p,q}(\mathbb{K})$ sont définies par :
  - la loi $+$: $\mathbf{A} + \mathbf{B} = (a_{i,j} + b_{i,j})_{i,j}$ vérifie les propriétés de la loi interne présentées dans la définition 1.4.
  - la loi $\cdot$: si $\lambda \in \mathbb{K}$, $\lambda \cdot \mathbf{B} = (\lambda \cdot b_{i,j})_{i,j}$ vérifie les propriétés de la loi externe de l'espace vectoriel.

1.3 Corps finis

Les corps finis sont des structures algébriques utilisés dans le domaine du codage. Un corps fini est **un corps**, une notion définie dans la définition 1.3, qui comporte un nombre fini d'éléments.

Théorème 1.1. *(Wedderburn) Tout corps fini est commutatif.*

La caractéristique d'un corps est le nombre de fois qu'il faut ajouter l'élément neutre de la loi multiplicative pour obtenir l'élément neutre de la loi additive. En d'autres termes, c'est le plus petit entier naturel $p \neq 0$ tel que $p \cdot 1 = 0$. Si un tel entier existe, sinon la caractéristique est dite nulle. Les corps $\mathbb{R}$ et $\mathbb{C}$ sont des corps de caractéristique nulle car un tel entier non nul n'existe pas.

Proposition 1.1. *Soit $\mathbb{K}$ un corps fini de caractéristique p avec $p \neq 0$ alors :*

- *p est premier,*
- *le plus petit corps de caractéristique p est $\mathbb{Z}/p\mathbb{Z} = \{0, 1, \cdots, p-1\}$,*
- *$\mathbb{K}$ est un espace vectoriel avec scalaire dans $\mathbb{Z}/p\mathbb{Z}$,*
- *$\mathbb{K}$ a exactement p^m éléments tel que m est sa dimension comme un $\mathbb{Z}/p\mathbb{Z}$-espace vectoriel.*

Un corps fini ayant p^m éléments est noté $\mathbb{F}_{p^m}$. Le corps $\mathbb{F}_{p^m}$ est appelé une extension finie de $\mathbb{F}_p$ où $\mathbb{F}_p$ est le corps de base. Il est également appelé un **corps de Galois** de cardinal p^m et noté $\mathrm{GF}(p^m)$. Par la suite, nous utiliserons cette dernière notation qui est celle la plus couramment utilisée dans le domaine de codage de canal.

1.3.1 Endomorphisme de Frobenius

L'endomorphisme de Frobenius est un objet fondamental pour étudier les corps finis. Il est défini dans la proposition 1.2.

Proposition 1.2. *Soit $\mathbb{K}$ un corps fini de caractéristique p alors l'endomorphisme de Frobenius :*

$$\begin{aligned} \sigma \quad : \quad & \mathbb{K} \to \mathbb{K} \\ & x \longmapsto x^p \end{aligned}$$

est un automorphisme de corps.

D'après cette proposition, pour tout a et b de $\mathbb{K}$, $\sigma(a+b) = \sigma(a) + \sigma(b)$ et $\sigma(a \cdot b) = \sigma(a) \cdot \sigma(b)$.

1.3.2 Propriétés des corps finis

Définition 1.5. *L'ordre d'un élément β non nul dans un corps fini est le plus petit entier $r \geq 1$ tel que $\beta^r = 1$.*

Théorème 1.2. *(de l'élément primitif) Tout corps fini $\mathbb{F}$ à p^m éléments contient au moins un élément β d'ordre $p^m - 1$. Un tel élément est appelé **élément primitif**.*

De ce théorème découle le corollaire suivant :

Corollaire 1.3. *Tout corps fini de taille p^m est de la forme GF(p^m)= $\{0, 1, \beta, \beta^2, \cdots, \beta^{p^m-2}\}$, pour un élément β convenable.*

Exemple 1.5.

Le corps fini de taille 2^3 est de la forme $\mathrm{GF}(2^3) = \{0, 1, \beta, \beta^2, \beta^3, \beta^4, \beta^5, \beta^6\}$

Proposition 1.4. *Soit $\mathbb{K}$ un corps fini à $q = p^m$ éléments où p est premier alors :*

- *tout élément x de $\mathbb{K}$ vérifie $x^q = x$,*
- *$\mathbb{K}$ est le corps de décomposition de $x^q - x$, c'est-à-dire que $x^q - x$ a q racines qui sont exactement les éléments de $\mathbb{K}$.*

On peut déduire de cette proposition que le corps fini $\mathbb{K}$ à p^m éléments, avec m un entier non nul et p premier, est un ensemble $\mathbb{K} = \{x,\ x^q - x = 0\}$. Il y a donc un et un seul corps à p^m éléments à isomorphisme de corps près.

1.3.3 Les polynômes irréductibles et polynômes primitifs

Un polynôme est dit irréductible s'il n'est divisible par aucun polynôme de degré inférieur.

Théorème 1.3. *Si $q = p^m$ où p est premier et m est un entier non nul alors :*
$x^q - x$ est le produit de tous les polynômes irréductibles dans $\mathbb{Z}/p\mathbb{Z}[x]$ de degré divisant m.

Nous mentionnons que les signes $+$ et $-$ sont équivalentes dans les corps de Galois de caractéristique 2, $\mathrm{GF}(2^m)$, c'est-à-dire si on a $x \in \mathrm{GF}(2^m)$ et $y \in \mathrm{GF}(2^m)$, alors $x + y = x - y$.

Exemple 1.6.

Dans cet exemple, nous cherchons tous les polynômes irréductibles dans les corps $\mathrm{GF}(2^2)$, $\mathrm{GF}(2^3)$, $\mathrm{GF}(2^4)$ et $\mathrm{GF}(2^5)$ en appliquant le Théorème 1.3.

Corps $\mathbf{GF(2^2)}$: Le polynôme $x^4 - x$ peut être factorisé en produit de tous les polynômes irréductibles de degré divisant 2. Les diviseurs de 2 sont 1 et 2, alors :

$$\begin{aligned} x^4 - x &= \underbrace{x \cdot (x+1)}_{\text{irréductibles de degré 1}} \\ &\cdot \underbrace{(x^2 + x + 1)}_{\text{irréductible de degré 2}} \end{aligned}$$

Corps $\mathbf{GF(2^3)}$: Le polynôme $x^8 - x$ peut être factorisé en produit de tous les polynômes irréductibles de degré divisant 3. Les diviseurs de 3 sont 1 et 3, alors :

$$\begin{aligned} x^8 - x &= \underbrace{x \cdot (x+1)}_{\text{irréductibles de degré 1}} \\ &\cdot \underbrace{(x^3 + x^2 + 1) \cdot (x^3 + x + 1)}_{\text{irréductibles de degré 3}} \end{aligned}$$

Corps $\mathbf{GF(2^4)}$: Le corps $\mathrm{GF}(2^4)$ est l'ensemble des racines de $x^{2^4} - x = x^{16} - x$. Pour factoriser le polynôme $x^{16} - x$ en produit de tous les polynômes irréductibles de degré divisant 4, le théorème 1.3 peut être utilisé. Les diviseurs de 4 sont 1, 2 et 4. Alors :

$$\begin{aligned} x^{16} - x &= \underbrace{x \cdot (x+1)}_{\text{irréductibles de degré 1}} \\ &\cdot \underbrace{(x^2 + x + 1)}_{\text{irréductible de degré 2}} \\ &\cdot \underbrace{(x^4 + x^3 + x^2 + x + 1) \cdot (x^4 + x^3 + 1) \cdot (x^4 + x + 1)}_{\text{irréductibles de degré 4}} \end{aligned}$$

Corps GF(2^5) : Le corps GF(2^5) est l'ensemble des racines de $x^{2^5} - x = x^{32} - x$. Le polynôme $x^{32} - x$ peut être factorisé en produit de tous les polynômes irréductibles de degré divisant 5. Les diviseurs de 5 sont 1 et 5. Alors :

$$\begin{aligned} x^{32} - x &= \underbrace{x \cdot (x+1)}_{\text{irréductibles de degré 1}} \\ &\cdot \underbrace{(x^5 + x^4 + x^2 + x + 1) \cdot (x^5 + x^3 + x^2 + x + 1)}_{\text{irréductibles de degré 5}} \\ &\cdot \underbrace{(x^5 + x^4 + x^3 + x + 1) \cdot (x^5 + x^4 + x^3 + x^2 + 1)}_{\text{irréductibles de degré 5}} \\ &\cdot \underbrace{(x^5 + x^3 + 1) \cdot (x^5 + x^2 + 1)}_{\text{irréductibles de degré 5}} \end{aligned}$$

Soit Q un polynôme irréductible dans $\mathbb{Z}/p\mathbb{Z}[x]$ de degré divisant m, $\mathbb{F} = \mathbb{Z}/p\mathbb{Z}[x]/Q$ est un corps car Q est irréductible. Tous les calculs se font avec les restes de la division des polynômes par Q.

Définition 1.6. ***Un polynôme primitif*** *est un polynôme qui admet une racine qui est un élément primitif.*

Il faut noter que tous les polynômes primitifs sont irréductibles. Mais la réciproque n'est pas toujours vraie car certains polynômes irréductibles ne sont pas primitifs. D'autre part, toute racine d'une polynôme primitif est un élément primitif.

Proposition 1.5. *Un polynôme irréductible $Q(x) \in \mathbb{F}_p[x]$ de degré m est dit primitif lorsque le plus petit entier positif n pour lequel $Q(x)$ divise $x^n - 1$ est $n = p^m - 1$.*

Exemple 1.7.

Soit $Q(x) = x^2+x+1$. C'est le seul polynôme irréductible dans GF(2) de degré 2. On va démontrer que ce polynôme est primitif.
Soit α une racine de $Q(x)$, alors on a $\alpha^2 + \alpha + 1 = 0$.

- $\alpha^0 = 1$
- $\alpha^1 = \alpha$
- $\alpha^2 = \alpha + 1$
- $\alpha^3 = \alpha \cdot (1 + \alpha) = \alpha + \alpha^2 = \alpha + \alpha + 1 = 1$

Donc α est un élément primitif qui génère tous les éléments non nuls du corps GF(2^2) par multiplication. D'où $Q(x) = x^2 + x + 1$ est un polynôme primitif.

Théorème 1.4. *Pour tout degré m, il existe toujours un polynôme primitif parmi les polynômes irréductibles dans $\mathbb{F}_p[x]$.*

Pour rechercher les polynômes primitifs du corps GF(p^m), le théorème 1.5 peut être appliqué.

Théorème 1.5. *Un élément de GF(p^m) est primitif si et seulement s'il est racine du polynôme :*

$$\phi(x) = \frac{x^{p^m-1} - 1}{\left(\prod_{d/(p^m-1) et\ d \neq p^m-1} (x^d - 1)\right) \wedge (x^{p^m-1} - 1)}$$

C'est-à-dire le polynôme $\phi(x)$ est le produit de tous les polynômes primitifs de degré m.

Exemple 1.8.

On veut chercher le nombre des polynômes primitifs dans les corps GF(2^2), GF(2^3), GF(2^4) et GF(2^5). Ce nombre correspond au degré $(\phi(x))/m$ où le degré de $\phi(x)$ peut être déterminé en utilisant le théorème 1.5.

Corps GF(2^2) : Les diviseurs de $2^2 - 1 = 3$ sont 1 et 3. En appliquant le théorème 1.5, un élément primitif β du corps GF(2^2) n'est pas une racine du polynôme $x - 1$. Alors, les polynômes primitifs sont les facteurs irréductibles de :

$$\phi(x) = \frac{x^3 - 1}{x - 1} = x^2 + x + 1$$

On peut déduire que le corps GF(2^2) possède un seul polynôme primitif de degré 2 qui est $x^2 + x + 1$. Nous remarquons que le polynôme irréductible de degré 2 trouvé dans l'exemple 1.6 correspond au polynôme primitif du corps GF(2^2).

Corps GF(2^3) : Les diviseurs de $2^3 - 1 = 7$ sont 1 et 7. En appliquant le théorème 1.5, un élément primitif β du corps GF(2^3) n'est pas une racine du polynôme $x - 1$. Alors, les polynômes primitifs sont les facteurs irréductibles de :

$$\phi(x) = \frac{x^7 - 1}{x - 1}$$

Le polynôme $x^7 - 1$ est de degré 7 et le polynôme $x - 1$ est de degré 1. Donc le degré du quotient de ces deux polynômes est $7 - 1 = 6 = 2 \cdot 3 = 2 \cdot m$. On peut déduire que le corps GF(2^3) possède deux polynômes primitifs. Nous remarquons que tous les polynômes irréductibles de degré 3 trouvés dans l'exemple 1.6 correspondent aux polynômes primitifs du corps GF(2^3).

Corps GF(2^4) : Les diviseurs de $2^4 - 1 = 15$ sont 1, 3, 5 et 15. En appliquant le théorème 1.5, un élément primitif β du corps GF(2^4) n'est pas une racine des polynômes :

- $x^3 - 1$,
- $x^5 - 1$,
- $(x^3 - 1) \vee (x^5 - 1)$.

Or, on a :

- $(x^3 - 1) \wedge (x^5 - 1) = x - 1$,
- $(x^3 - 1) \vee (x^5 - 1) = \dfrac{(x^3 - 1).(x^5 - 1)}{x - 1} = (x^2 + x + 1) \cdot (x^5 - 1)$

Alors, les polynômes primitifs sont les facteurs irréductibles de :

$$\phi(x) = \frac{x^{15} - 1}{(x^3 - 1) \vee (x^5 - 1)}$$

Le polynôme $x^{15} - 1$ est de degré 15 et le polynôme $(x^3 - 1) \vee (x^5 - 1) = (x^2 + x + 1) \cdot (x^5 - 1)$ est de degré $2 + 5 = 7$. Donc le degré du quotient de ces deux polynômes est $15 - 7 = 8 = 2 \cdot 4 = 2 \cdot m$. On peut déduire que le corps GF(2^4) possède deux polynômes primitifs. Nous remarquons que parmi les polynômes irréductibles de degré 4 trouvés dans l'exemple 1.6 il

n'existe que deux polynômes primitifs du corps GF(2^4).

Corps GF(2^5) : Les diviseurs de $2^5 - 1 = 31$ sont 1 et 31. En appliquant le théorème 1.5, un élément primitif β du corps GF(2^5) n'est pas une racine du polynôme $x - 1$. Alors, les polynômes primitifs sont les facteurs irréductibles de :

$$\phi(x) = \frac{x^{31} - 1}{x - 1}$$

Le polynôme $x^{31} - 1$ est de degré 31 et le polynôme $x - 1$ est de degré 1. Donc le degré du quotient de ces deux polynômes est $31 - 1 = 30 = 6 \cdot 5 = 6 \cdot m$. On peut déduire que le corps GF(2^5) possède six polynômes primitifs. Nous remarquons que tous les polynômes irréductibles de degré 5 trouvés dans l'exemple 1.6 correspondent aux polynômes primitifs du corps GF(2^5).

1.3.4 Le polynôme minimal

Définition 1.7. *Étant donné un élément β appartenant à GF(p^m), le* ***polynôme minimal*** *de β est un polynôme de plus petit degré à coefficients dans GF(p) ayant β comme racine.*

Théorème 1.6. *Pour chaque élément β dans GF(p^m), il existe un unique polynôme minimal $P(x)$ à coefficients dans GF(p) tel que :*

- $P(\beta) = 0$,
- *le degré de* $P(x) \leq m$,
- *s'il existe un polynôme* $f(x) \in GF(p)[x]$ *tel que* $f(\beta) = 0$ *alors* $P(x)$ *divise* $f(x)$,
- $P(x)$ *est irréductible dans GF(p)*$[x]$.

Théorème 1.7. *Dans un corps fini de dimension p^m, les polynômes primitifs sont les polynômes minimaux de leurs racines (éléments primitifs).*

Définition 1.8. *Soit $\beta \in$ GF(p^m), les conjugués de β sont :*

$$\beta, \sigma(\beta) = \beta^p, \sigma^2(\beta) = \beta^{p^2}, \sigma^3(\beta) = \beta^{p^3}, \cdots, \sigma^{d-1}(\beta) = \beta^{p^{d-1}}$$

tel que $\beta^{p^d} = \beta$. Cet ensemble de conjugués forme un ensemble appelé classe conjuguée de β.

Théorème 1.8. *Le polynôme minimal d'un élément β se décompose sous la forme :*

$$M_\beta(x) = (x - \beta) \cdot (x - \beta^p) \cdots (x - \beta^{p^{d-1}}) = \prod_{0 \leq i < d} (x - \sigma^i(\beta)) \tag{1.1}$$

tel que l'ensemble $\{\beta, \beta^p, \cdots, \beta^{p^{d-1}}\}$ est la classe conjuguée de β.

Exemple 1.9.

Dans cet exemple, notre objectif est de chercher les polynômes minimaux d'éléments des corps GF(2^2), GF(2^3), GF(2^4) et GF(2^5) en utilisant la définition 1.8 et le théorème 1.8.

Corps GF(2^2) : Soit α un élément primitif dans GF(2^2)$= \{0, 1, \alpha, \alpha^2\}$. On veut déterminer les classes conjuguées dans GF(2^2) afin de déterminer leurs polynômes minimaux.

- On choisit l'élément α dont la liste des conjugués est donnée par :

$$\alpha, \sigma(\alpha) = \alpha^2, \sigma^2(\alpha) = \alpha^4 = \alpha$$

alors la première classe conjuguée de $\mathrm{GF}(2^2)$ est $\{\alpha, \alpha^2\}$. Le polynôme minimal qui correspond à cette classe est donné par :

$$M_\alpha(x) = (x - \alpha) \cdot (x - \alpha^2) = x^2 + x + 1$$

- On choisit un autre élément de $\mathrm{GF}(2^2)$ non utilisé dans la première classe. Il reste les classes de 0 et 1 qui sont respectivement $\{0\}$ et $\{1\}$. Leurs polynômes minimaux sont donnés par :

$$M_0(x) = x \tag{1.2}$$
$$M_1(x) = x + 1 \tag{1.3}$$

Le seul polynôme irréductible de degré 2 dans $\mathbb{Z}/2\mathbb{Z}$ est donc $M_\alpha(x) = x^2 + x + 1$. D'après le théorème 1.7, $M_\alpha(x)$ est un polynôme primitif puisque α est sa racine.

Corps GF(2^3) : Soit α une racine de polynôme primitif $x^3 + x + 1$ dans $\mathrm{GF}(2^3)= \{0, 1, \alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6\}$. On veut déterminer les classes conjuguées dans $\mathrm{GF}(2^3)$ afin de déterminer leurs polynômes minimaux.

- La liste des conjugués de α est déterminée par :

$$\alpha, \sigma(\alpha) = \alpha^2, \sigma^2(\alpha) = \alpha^4, \sigma^3(\alpha) = \alpha^8 = \alpha$$

alors la première classe conjuguée de $\mathrm{GF}(2^3)$ est $\{\alpha, \alpha^2, \alpha^4\}$. Le polynôme minimal qui correspond à cette classe est donné par :

$$M_\alpha(x) = (x - \alpha) \cdot (x - \alpha^2) \cdot (x - \alpha^4) = x^3 + x + 1$$

- On choisit un autre élément de $\mathrm{GF}(2^3)$ non utilisé dans la première classe. On prend α^3 qui a comme liste de conjuguées :

$$\alpha^3, \sigma(\alpha^3) = (\alpha^3)^2 = \alpha^6, \sigma^2(\alpha^3) = \alpha^{12} = \alpha^5, \sigma^3(\alpha^3) = \alpha^{24} = \alpha^3$$

la classe conjuguée de α^3 est $\{\alpha^3, \alpha^6, \alpha^5\}$. Le polynôme minimal qui correspond à cette classe est donné par :

$$M_{\alpha^3}(x) = (x - \alpha^3) \cdot (x - \alpha^5) \cdot (x - \alpha^6) = x^3 + x^2 + 1$$

- Il reste les classes de 0 et 1 qui sont respectivement $\{0\}$ et $\{1\}$. Leurs polynômes minimaux sont donnés respectivement par les équations (1.2) et (1.3).

Les polynômes irréductibles de degré 3 dans $\mathbb{Z}/2\mathbb{Z}$ sont donc $x^3 + x + 1$ et $x^3 + x^2 + 1$.

Corps GF(2^4) : Soit α une racine de polynôme primitif x^4+x+1 dans le corps $\mathrm{GF}(2^4)= \{0\} \cup \{\alpha^i,\ i \in [\![0, 14]\!]\}$. On veut déterminer les classes conjuguées dans $\mathrm{GF}(2^4)$ afin de déterminer leurs polynômes minimaux.

- La liste des conjugués de α est donnée par :

$$\alpha, \sigma(\alpha) = \alpha^2, \sigma^2(\alpha) = \alpha^4, \sigma^3(\alpha) = \alpha^8, \sigma^4(\alpha) = \alpha^{16} = \alpha$$

alors la première classe conjuguée de GF(2^4) est $\{\alpha, \alpha^2, \alpha^4, \alpha^8\}$. Le polynôme minimal qui correspond à cette classe est donné par :

$$M_\alpha(x) = (x-\alpha)\cdot(x-\alpha^2)\cdot(x-\alpha^4)\cdot(x-\alpha^8) = x^4+x+1$$

- On choisit un autre élément de GF(2^4) non utilisé dans la première classe. On prend α^3 qui a comme liste de conjuguées :

$$\alpha^3, \sigma(\alpha^3) = \alpha^6, \sigma^2(\alpha^3) = \alpha^{12}, \sigma^3(\alpha^3) = \alpha^{24} = \alpha^9, \sigma^4(\alpha^3) = \alpha^{48} = \alpha^3$$

  la classe conjuguée de α^3 est $\{\alpha^3, \alpha^6, \alpha^9, \alpha^{12}\}$. Le polynôme minimal qui correspond à cette classe est donné par :

$$M_{\alpha^3}(x) = (x-\alpha^3)\cdot(x-\alpha^6)\cdot(x-\alpha^9)\cdot(x-\alpha^{12}) = x^4+x^3+x^2+x+1$$

- On choisit un deuxième élément de GF(2^4) non utilisé dans les classes de α et α^3. On prend α^5 qui a comme liste de conjuguées :

$$\alpha^5, \sigma(\alpha^5) = \alpha^{10}, \sigma^2(\alpha^5) = \alpha^{20} = \alpha^5$$

  la classe conjuguée de α^5 est $\{\alpha^5, \alpha^{10}\}$. Le polynôme minimal qui correspond à cette classe est donné par :

$$M_{\alpha^5}(x) = (x-\alpha^5)\cdot(x-\alpha^{10}) = x^2+x+1$$

- Les conjugués de α^7 sont :

$$\alpha^7, \sigma(\alpha^7) = \alpha^{14}, \sigma^2(\alpha^7) = \alpha^{28} = \alpha^{13}, \sigma^3(\alpha^7) = \alpha^{11}, \sigma^4(\alpha^7) = \alpha^7$$

  la classe conjuguée de cet élément est $\{\alpha^7, \alpha^{11}, \alpha^{13}, \alpha^{14}\}$. Le polynôme minimal qui correspond à cette classe est donné par :

$$M_{\alpha^7}(x) = (x-\alpha^7)\cdot(x-\alpha^{11})\cdot(x-\alpha^{13})\cdot(x-\alpha^{14}) = x^4+x^3+1$$

- Il reste les classes de 0 et 1 qui sont respectivement $\{0\}$ et $\{1\}$. Leurs polynômes minimaux sont donnés respectivement par les équations (1.2) et (1.3).

Les polynômes irréductibles de degré 4 dans $\mathbb{Z}/2\mathbb{Z}$ sont donc x^4+x+1, $x^4+x^3+x^2+x+1$ et x^4+x^3+1.

Corps GF(2^5) : Soit α une racine de polynôme primitif x^5+x^2+1 dans GF(2^5)$= \{0\}\cup\{\alpha^i,\ i \in [\![0,30]\!]\}$. On veut déterminer les classes conjuguées dans GF(2^5) afin de déterminer leurs polynômes minimaux.

- On choisit α un élément du corps et on liste ses conjugués :

$$\alpha, \sigma(\alpha) = \alpha^2, \sigma^2(\alpha) = \alpha^4, \sigma^3(\alpha) = \alpha^8, \sigma^4(\alpha) = \alpha^{16}, \sigma^5(\alpha) = \alpha^{32} = \alpha$$

  alors la première classe conjuguée de GF(2^5) est $\{\alpha, \alpha^2, \alpha^4, \alpha^8, \alpha^{16}\}$. Le polynôme minimal qui correspond à cette classe est donné par :

$$M_\alpha(x) = (x-\alpha)\cdot(x-\alpha^2)\cdot(x-\alpha^4)\cdot(x-\alpha^8)\cdot(x-\alpha^{16}) = x^5+x^2+1$$

- On choisit un deuxième élément de GF(2^5) non utilisé dans la première classe. On prend

α^3 qui a comme liste de conjuguées :

$$\alpha^3, \sigma(\alpha^3) = \alpha^6, \sigma^2(\alpha^3) = \alpha^{12}, \sigma^3(\alpha^3) = \alpha^{24}, \sigma^4(\alpha^3) = \alpha^{17}, \sigma^5(\alpha^3) = \alpha^3$$

la deuxième classe conjuguée est $\{\alpha^3, \alpha^6, \alpha^{12}, \alpha^{24}, \alpha^{17}\}$. Le polynôme minimal qui correspond à cette classe est donné par :

$$M_{\alpha^3}(x) = (x - \alpha^3) \cdot (x - \alpha^6) \cdot (x - \alpha^{12}) \cdot (x - \alpha^{24}) \cdot (x - \alpha^{17}) = x^5 + x^4 + x^3 + x^2 + 1$$

- On prend un autre élément du corps non utilisé comme α^5. Ses conjugués sont :

$$\alpha^5, \sigma(\alpha^5) = \alpha^{10}, \sigma^2(\alpha^5) = \alpha^{20}, \sigma^3(\alpha^5) = \alpha^9, \sigma^4(\alpha^5) = \alpha^{18}, \sigma^5(\alpha^5) = \alpha^5$$

la classe conjuguée de α^5 est $\{\alpha^5, \alpha^{10}, \alpha^{20}, \alpha^9, \alpha^{18}\}$. Le polynôme minimal qui correspond à cette classe est donné par :

$$M_{\alpha^5}(x) = (x - \alpha^5) \cdot (x - \alpha^{10}) \cdot (x - \alpha^{20}) \cdot (x - \alpha^9) \cdot (x - \alpha^{18}) = x^5 + x^4 + x^3 + x^2 + 1$$

- les conjugués de α^7 sont :

$$\alpha^7, \sigma(\alpha^7) = \alpha^{14}, \sigma^2(\alpha^7) = \alpha^{28}, \sigma^3(\alpha^7) = \alpha^{25}, \sigma^4(\alpha^7) = \alpha^{19}, \sigma^5(\alpha^7) = \alpha^7$$

la classe conjuguée de cet élément est $\{\alpha^7, \alpha^{14}, \alpha^{28}, \alpha^{25}, \alpha^{19}\}$. Le polynôme minimal qui correspond à cette classe est donné par :

$$M_{\alpha^7}(x) = (x - \alpha^7) \cdot (x - \alpha^{14}) \cdot (x - \alpha^{28}) \cdot (x - \alpha^{25}) \cdot (x - \alpha^{19}) = x^5 + x^3 + x^2 + x + 1$$

- les conjugués de α^{11} sont :

$$\alpha^{11}, \sigma(\alpha^{11}) = \alpha^{22}, \sigma^2(\alpha^{11}) = \alpha^{13}, \sigma^3(\alpha^{11}) = \alpha^{26}, \sigma^4(\alpha^{11}) = \alpha^{21}, \sigma^5(\alpha^{11}) = \alpha^{11}$$

la classe conjuguée de cet élément est $\{\alpha^{11}, \alpha^{22}, \alpha^{13}, \alpha^{26}, \alpha^{21}\}$. Le polynôme minimal qui correspond à cette classe est donné par :

$$M_{\alpha^{11}}(x) = (x - \alpha^{11}) \cdot (x - \alpha^{22}) \cdot (x - \alpha^{13}) \cdot (x - \alpha^{26}) \cdot (x - \alpha^{21}) = x^5 + x^4 + x^3 + x + 1$$

- les conjugués de α^{15} sont :

$$\alpha^{15}, \sigma(\alpha^{15}) = \alpha^{30}, \sigma^2(\alpha^{15}) = \alpha^{29}, \sigma^3(\alpha^{15}) = \alpha^{27}, \sigma^4(\alpha^{15}) = \alpha^{23}, \sigma^5(\alpha^{15}) = \alpha^{15}$$

la classe conjuguée de cet élément est $\{\alpha^{15}, \alpha^{30}, \alpha^{29}, \alpha^{27}, \alpha^{23}\}$. Le polynôme minimal qui correspond à cette classe est donné par :

$$M_{\alpha^{15}}(x) = (x - \alpha^{15}) \cdot (x - \alpha^{30}) \cdot (x - \alpha^{29}) \cdot (x - \alpha^{27}) \cdot (x - \alpha^{23}) = x^5 + x^3 + 1$$

- Il reste les classes de 0 et 1 qui sont respectivement $\{0\}$ et $\{1\}$.

Les polynômes irréductibles de degré 5 dans $\mathbb{Z}/2\mathbb{Z}$ sont donc x^5+x^2+1, $x^5+x^4+x^3+x^2+1$, $x^5 + x^4 + x^2 + x + 1$, $x^5 + x^3 + x^2 + x + 1$, $x^5 + x^4 + x^3 + x + 1$ et $x^5 + x^3 + 1$.

1.4 Construction d'un corps fini de caractéristique 2

1.4.1 Principe de la méthode de construction

Nous introduisons un corps de base ayant 2 éléments $\{0, 1\}$. Il s'agit du corps de Galois binaire noté GF(2). Les opérations dans ce corps sont effectuées modulo 2. Notre objectif est de construire un corps qui admet 2^m éléments, noté GF(2^m). De ce fait, nous donnons une méthode générale de construction. Cette méthode est basée sur les notions définies dans la section 1.3 qui introduit les corps finis.

La construction d'un corps GF(2^m) utilise un polynôme primitif de degré m dont une racine permet de générer tous les éléments du corps. Pour cela, il faut, tout d'abord, chercher tous les polynômes irréductibles sur GF(2) de degré m. Puis, on détermine parmi ces polynômes ceux qui ont les caractéristiques d'un polynôme primitif. Pour un corps de cardinal 2^m, il peut y avoir un ou plusieurs polynômes primitifs qui permet(tent) de le construire. Les éléments d'un corps de Galois possèdent différentes représentations suivant le polynôme primitif utilisé pour construire ce corps. Un tableau récapitulatif de différents polynômes minimaux, polynômes irréductibles et polynômes primitifs pour les corps GF(2^m), avec $m = \{2, 3, 4, 5\}$, est présenté dans l'annexe A. Dans cette section, nous illustrons la méthode de construction du corps GF(2^4).

1.4.2 Construction du corps GF(2^4)

1.4.2.1 Recherche des polynômes irréductibles sur GF(2^4)

Un polynôme irréductible de degré 4 a nécessairement un nombre impair de termes dont le terme 1 est inclus puisque ni 0 ni 1 ne sont racines. Dans notre cas, les candidats pouvant être des polynômes irréductibles sont :

- $P_1(x) = x^4 + x^3 + x^2 + x + 1$
- $P_2(x) = x^4 + x^3 + 1$
- $P_3(x) = x^4 + x^2 + 1$
- $P_4(x) = x^4 + x + 1$

En appliquant l'endomorphisme de Frobenius, défini dans la proposition 1.2, au polynôme x^2+x+1, $\sigma(x^2 + x + 1) = (x^2 + x + 1)^2 = x^4 + x^2 + 1 = P_3(x)$, on démontre que $P_3(x)$ n'est pas irréductible puisqu'il est un produit de deux polynômes irréductibles de degré 2. Donc, seuls 3 polynômes parmi les 4 sont ni facteur de degré 1 ni facteur de degré 2. Ils sont donc irréductibles.

1.4.2.2 Recherche des polynômes primitifs

Nous vérifions si les polynômes irréductibles $P_1(x)$, $P_2(x)$ et $P_4(x)$ peuvent générer tous les éléments du corps GF(2^4). Si c'est le cas, alors ce polynôme est un polynôme primitif.

- Le polynôme irréductible $P_1(x)$ est-t-il primitif ?

Soit β la classe de x dans $\mathbb{F}_2[x]/P_1(x)$ tel que $\beta^4 + \beta^3 + \beta^2 + \beta + 1 = 0$. Nous vérifions si β est d'ordre 15 pour être un élément primitif. En effet, $\beta^5 - 1$ peut être factorisé en :

$$\beta^5 - 1 = (\beta - 1) \cdot (\beta^4 + \beta^3 + \beta^2 + \beta + 1)$$

et puisque $\beta^4 + \beta^3 + \beta^2 + \beta + 1 = 0$, alors $\beta^5 - 1 = 0$ ce qui implique $\beta^5 = 1$, c'est-à-dire l'ordre de β est $5 \neq 2^4 - 1 = 15$. D'après le théorème 1.2, β n'est pas un élément primitif. Donc le polynôme $P_1(x)$ n'est pas primitif.

- Le polynôme irréductible $P_2(x)$ est-t-il primitif ?

Soit β la classe de x dans $\mathbb{F}_2[x]/P_2(x)$ tel que $\beta^4 + \beta^3 + 1 = 0$. On dit que β est primitif s'il est d'ordre 15 dans le groupe ($\mathrm{GF}(2^4) \setminus \{0\}, \cdot$).

Les diviseurs de 15 sont 1, 3, 5 et 15, alors β est primitif $\Longleftrightarrow \beta^3 \neq 1$ et $\beta^5 \neq 1$. Or, on a par construction $\beta^3 \neq 1$. Vérifions $\beta^5 = \beta \cdot \beta^4 = \beta \cdot (\beta^3 + 1) = \beta^4 + \beta = \beta^3 + 1 + \beta \neq 1$. Donc, β est un élément primitif et $P_2(x)$ est un polynôme primitif de $\mathrm{GF}(2^4)$.

- Le polynôme irréductible $P_4(x)$ est-t-il primitif ?

Soit β la classe de x dans $\mathbb{F}_2[x]/P_4(x)$ tel que $\beta^4+\beta+1 = 0$, on a $\beta^3 \neq 1$ et $\beta^5 = \beta.\beta^4 = \beta.(\beta+1) = \beta^2 + \beta \neq 1$. Alors β est un élément primitif et $P_4(x)$ est un polynôme primitif de $\mathrm{GF}(2^4)$.

1.4.2.3 Les différentes représentations des symboles du corps GF(2^4)

Le corps $\mathrm{GF}(2^4)$ admet 16 éléments. Ses éléments peuvent être représentés en fonction des racines des polynômes primitifs. Les polynômes primitifs $P_2(x)$ et $P_4(x)$ permettent de représenter les éléments du corps. Chaque polynôme possède une racine primitive notée $\beta_i, i \in \{2, 4\}$. Nous allons voir la différence entre ces deux polynômes dans la construction du corps. Soit $(\beta_i^3, \beta_i^2, \beta_i, 1)$ la base utilisée pour construire le corps $\mathrm{GF}(2^4)$. Chaque élément du corps peut être représenté dans une base s'il est vu comme un espace vectoriel de dimension 4 sur $\mathrm{GF}(2)$ par un polynôme de degré au plus 3 avec $\beta_2^4 = \beta_2^3 + 1$ et $\beta_4^4 = \beta_4 + 1$. Les tableaux 1.2 et 1.3 résument les deux représentations en utilisant les racines β_2 et β_4 des deux polynômes primitifs $P_2(x)$ et $P_4(x)$.

1.5 Conclusion

Dans ce chapitre, nous avons tout d'abord présenté les différentes structures algébriques afin de traiter leurs constructions et leurs propriétés. Puis, nous nous sommes concentrés sur la structure du corps de Galois, qui est une structure très utilisée dans le domaine du codage canal. Nous avons vu quelques notions sur les corps finis qui nous permettent de construire un corps de Galois de cardinal 2^m qui fait l'objet de notre étude. Nous avons montré que cette construction d'un corps est basée sur la connaissance du cardinal du corps, plus précisément du paramètre m, et du polynôme primitif. Cette étude sur le corps de Galois nous a permis d'obtenir les paramètres indispensables pour générer les éléments non-binaires de ce corps. L'influence de ces paramètres sur les méthodes d'identification aveugle des codes correcteurs d'erreurs non-binaires sera étudiée dans les prochains chapitres.

$(\beta_2^3, \beta_2^2, \beta_2, 1)$	**Polynôme**	**Puissance de β_2**
$(0,0,0,0)$	0	0
$(0,0,0,1)$	1	$1 = \beta_2^0 = \beta_2^{15}$
$(0,0,1,0)$	β_2	β_2
$(0,1,0,0)$	β_2^2	β_2^2
$(1,0,0,0)$	β_2^3	β_2^3
$(1,0,0,1)$	$\beta_2^3 + 1$	β_2^4
$(1,0,1,1)$	$\beta_2^3 + \beta_2 + 1$	β_2^5
$(1,1,1,1)$	$\beta_2^3 + \beta_2^2 + \beta_2 + 1$	β_2^6
$(0,1,1,1)$	$\beta_2^2 + \beta_2 + 1$	β_2^7
$(1,1,1,0)$	$\beta_2^3 + \beta_2^2 + \beta_2$	β_2^8
$(0,1,0,1)$	$\beta_2^2 + 1$	β_2^9
$(1,0,1,0)$	$\beta_2^3 + \beta_2$	β_2^{10}
$(1,1,0,1)$	$\beta_2^3 + \beta_2^2 + 1$	β_2^{11}
$(0,0,1,1)$	$\beta_2 + 1$	β_2^{12}
$(0,1,1,0)$	$\beta_2^2 + \beta_2$	β_2^{13}
$(1,1,0,0)$	$\beta_2^3 + \beta_2^2$	β_2^{14}

Tableau 1.2 — Représentation des éléments du corps GF(2^4)= $\mathbb{F}_2[x]/P_2(x)$

$(\beta_4^3, \beta_4^2, \beta_4, 1)$	**Polynôme**	**Puissance de β_4**
$(0,0,0,0)$	0	0
$(0,0,0,1)$	1	$1 = \beta_4^0 = \beta_4^{15}$
$(0,0,1,0)$	β_4	β_4
$(0,1,0,0)$	β_4^2	β_4^2
$(1,0,0,0)$	β_4^3	β_4^3
$(0,0,1,1)$	$\beta_4 + 1$	β_4^4
$(0,1,1,0)$	$\beta_4^2 + \beta_4$	β_4^5
$(1,1,0,0)$	$\beta_4^3 + \beta_4^2$	β_4^6
$(1,0,1,1)$	$\beta_4^3 + \beta_4 + 1$	β_4^7
$(0,1,0,1)$	$\beta_4^2 + 1$	β_4^8
$(1,0,1,0)$	$\beta_4^3 + \beta_4$	β_4^9
$(0,1,1,1)$	$\beta_4^2 + \beta_4 + 1$	β_4^{10}
$(1,1,1,0)$	$\beta_4^3 + \beta_4^2 + \beta_4$	β_4^{11}
$(1,1,1,1)$	$\beta_4^3 + \beta_4^2 + \beta_4 + 1$	β_4^{12}
$(1,1,0,1)$	$\beta_4^3 + \beta_4^2 + 1$	β_4^{13}
$(1,0,0,1)$	$\beta_4^3 + 1$	β_4^{14}

Tableau 1.3 — Représentation des éléments du corps GF(2^4)= $\mathbb{F}_2[x]/P_4(x)$

CHAPITRE 2

Transmissions numériques et codes correcteurs d'erreurs

2.1 Introduction

L'introduction d'un système de codage performant à l'émission comme les codes correcteurs d'erreurs est indispensable pour combattre l'effet des perturbations introduites par le canal de transmission. La plupart des travaux de recherche se sont souvent restreints à des codes manipulant des données binaires. Pour les codes correcteurs d'erreurs non-binaires travaillant dans les corps de Galois $\mathrm{GF}(q)$, avec $q > 2$, les implémentations et les recherches associées se sont très longtemps limitées aux codes de Reed-Solomon.

L'objectif de ce chapitre est d'introduire les propriétés de codes linéaires dans $\mathrm{GF}(q)$ afin d'extraire les paramètres nécessaires pour les reconnaître à la réception. Mais, avant cela, nous présentons tout d'abord les différentes opérations utilisées dans une chaîne de transmission numérique ainsi que les différents modèles de canaux de transmission que nous abordons dans nos travaux.

2.2 Chaîne de transmission numérique

2.2.1 Principe

Le principe d'une chaîne de transmission classique est de transmettre une information sous forme numérique à partir d'une source vers un ou plusieurs destinataires. La structure de cette chaîne est représentée sur la figure 2.1. Elle est composée de plusieurs blocs décrits ci-dessous.

- La source d'information : elle convertit le message d'information de type analogique en une séquence d'éléments binaires appartenant à $\{0, 1\} = \mathrm{GF}(2)$.
- Le codeur de source : il cherche à transmettre uniquement l'information utile en éliminant toute redondance associée au message afin d'augmenter l'efficacité de la transmission et d'optimiser l'utilisation des ressources du système. Dans le cadre de cette thèse, nous ne nous intéresserons pas au codage de source. Nous supposerons que la source d'information délivre des données binaires indépendantes et identiquement distribuées.
- Le codeur de canal : il introduit de la redondance de manière contrôlée dans la séquence d'information afin de rendre la transmission des données utiles plus fiable.
- Le modulateur : il adapte la séquence des symboles générée par le codeur de canal au canal de transmission. En effet, une forme d'onde va être associée à chaque groupe d'éléments de

la séquence codée pour construire un signal électrique. Cette forme d'onde est plus souvent un signal porte pour les transmissions en bande de base ou un signal sinusoïdal pour les transmissions sur fréquences porteuses.

- Le canal de transmission : il s'agit d'un support physique sur lequel les symboles modulés seront transmis pour arriver au récepteur. Ce support va nécessairement plus ou moins perturber le signal transmis.
- Le démodulateur : il consiste à traiter les formes d'onde issues du canal de transmission afin de les traduire en symboles dans GF(q), où $q \geqslant 2$ (binaires ou non-binaires). La sortie du démodulateur peut être ferme ou souple. Lorsque le démodulateur fournit des symboles dans GF(q), une décision ferme a été prise. La décision souple permet de fournir des probabilités de vraisemblance du symbole reçu au symbole émis. Pour estimer la valeur du symbole reçu, une règle de maximum de vraisemblance (MAP) va être appliquée.
- Le décodeur de canal : il détecte et/ou corrige les erreurs de transmission en exploitant la redondance introduite par le codeur de canal.
- Le décodeur de source : il restitue le message d'origine.

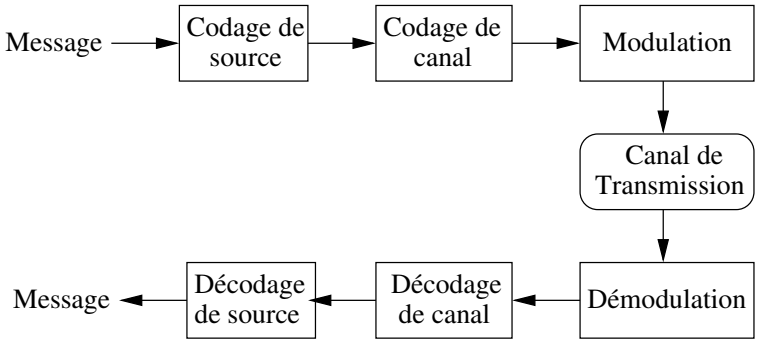


Figure 2.1 — Chaîne standard de transmission numérique

Dans le cadre de cette thèse, nous étudions la chaîne de transmission présentée dans la figure 2.2, et plus précisément le bloc de codage dans GF(q) et les blocs qui correspondent aux fonctions de mapping-demapping.

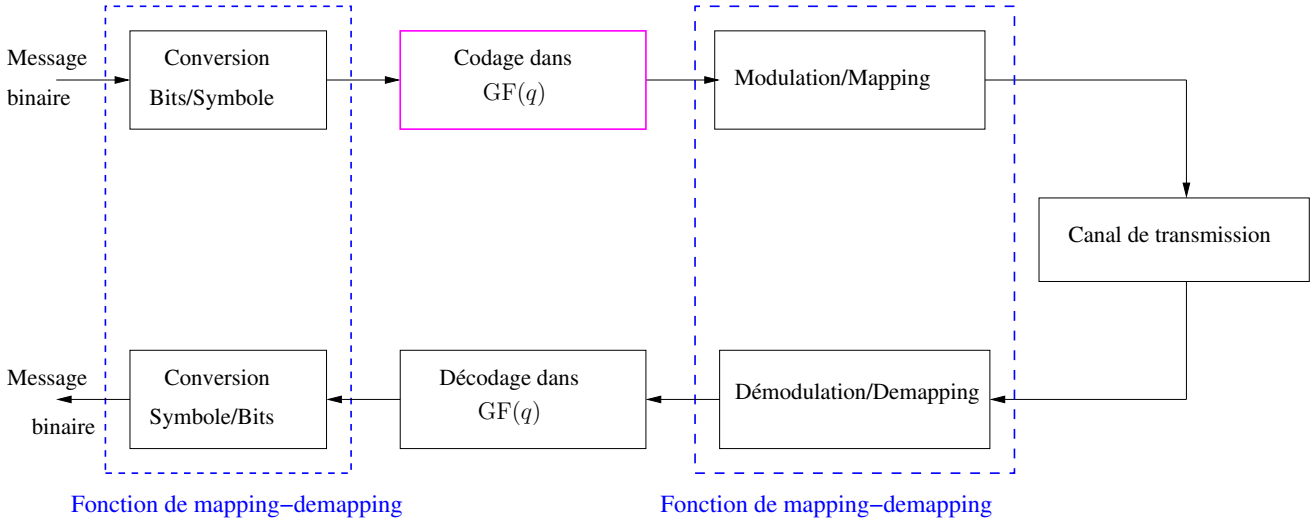


Figure 2.2 — Chaîne étudiée de transmission numérique

Cette chaîne comporte un codeur de canal qui traite les données binaires et non-binaires. Pour générer une séquences d'information non-binaire dans GF(q), une opération de conversion de chaque groupe d'un nombre déterminé de bits m (typiquement $q = 2^m$) en un symbole de GF(q) est indispensable. Le principe de cette opération a été illustré dans le premier chapitre. L'opération de conversion de bits en un symbole de GF(q) peut être vue comme une opération de mapping qui consiste à associer à un élément du corps de Galois une représentation binaire spécifique. Par contre, l'opération de conversion d'un symbole en un groupe de bits peut correspondre à une opération de demapping qui représente une opération inverse de mapping. D'un point de vue mathématique, la composition de ces deux opérations, appelée fonction de mapping-demapping, doit être égale à l'identité. L'opération de modulation est aussi modélisée par une opération de conversion bits/symbole, appelée aussi mapping. Dans le cas d'une modulation d'ordre $q = 2^m$, où q représente le nombre de points de la constellation associée à la modulation, cette opération consiste à associer à chaque ensemble de m éléments binaires de l'information utile un symbole complexe. En général, la conversion se fait selon le mapping de Gray. La composition des deux opérations de modulation et de démodulation correspond aussi à une fonction de mapping-demapping.

Dans la première partie de ce mémoire, nous proposons des algorithmes d'identification aveugle des paramètres des codeurs dans GF(q). La deuxième partie est consacrée à l'étude de fonctions de mapping-demapping afin de voir l'impact d'utiliser un demapping inapproprié, qui ne correspond pas à l'opération inverse de mapping utilisé à l'émission, sur l'identification aveugle des codes non-binaires et sur la manipulation des éléments du corps de Galois. Dans ce chapitre, nous nous concentrons sur le bloc du codage canal, en particulier les codes correcteurs d'erreurs non-binaires. Mais avant cela, nous présentons les modèles de canaux de transmission utilisés dans nos travaux.

2.2.2 Modèles des canaux de transmission

Pour pouvoir récupérer l'information transmise à partir des données reçues, le récepteur doit être adapté aux caractéristiques du canal de transmission. Un canal de transmission est un milieu physique qui assure le lien entre la source et le destinataire. La forme physique du signal émis sera adaptée au milieu. Citons par exemple, un signal électrique pour les câbles, un signal optique pour les fibres ou un signal électromagnétique dans l'atmosphère. Lors de la transmission sur un canal de transmission, le signal émis subit des atténuations et des déformations qui affectent la qualité du signal. Par conséquent, le récepteur qui reçoit un signal bruité et distordu doit être en mesure de le corriger et de retrouver le signal d'origine. Du point de vue de la théorie de l'information, le canal de transmission reçoit en entrée des symboles issus du modulateur et délivre en sortie des symboles pouvant être entachés d'erreurs. Ces erreurs pourront en partie être corrigées à la réception en connaissant un modèle du canal de transmission utilisé pour les générer.

Afin d'évaluer la qualité d'une transmission, les critères de taux d'erreurs binaires, noté TEB, et de taux d'erreurs symboles, noté TES, sont fréquemment utilisés. Le critère de TEB est défini par le rapport entre le nombre d'éléments binaires erronés et le nombre d'éléments binaires transmis. Par contre, le TES représente le rapport entre le nombre de symboles erronés et le nombre de symboles transmis. Ces deux critères sont souvent calculés en fonction du rapport signal à bruit, noté RSB, qui représente le rapport entre la puissance du signal émis et la puissance du bruit ou en fonction du rapport E_b/N_0. Ce dernier est définit par le rapport entre l'énergie moyenne par bit, noté E_b, et la densité spectrale de puissance du bruit, notée N_0. Le rapport E_s/N_0 qui est définit par le rapport entre l'énergie de transmission d'un symbole, noté E_s, et N_0 peut être également utilisé pour mesurer le TEB et le TES.

2.2.2.1 Canal binaire symétrique

C'est le modèle de canal le plus simple. Ce modèle, aussi appelé BSC ("Binary Symmetric Channel"), est un canal discret sans mémoire dont les entrées, représentées par le vecteur $\mathbf{c}$, et les sorties, représentées par le vecteur $\mathbf{r}$, sont des symboles binaires dans GF(2)= $\{0, 1\}$. Les erreurs générées par ce canal sont des variables aléatoires indépendantes qui suivent toutes une loi de Bernoulli de même paramètre p_e. En particulier, le bit transmis c_i, $\forall i \in [\![1, L]\!]$, avec L la longueur de la séquence transmise, est reçu erroné avec une probabilité p_e et correctement avec une probabilité $1 - p_e$:

$$\begin{aligned} p(r_i = 0|c_i = 1) &= p(r_i = 1|c_i = 0) = p_e \\ p(r_i = 0|c_i = 0) &= p(r_i = 1|c_i = 1) = 1 - p_e \end{aligned} \tag{2.1}$$

Le fonctionnement d'un BSC est décrit sur la figure 2.3. Nous précisons que le bloc du modulateur, canal de transmission et démodulateur est équivalent à un canal binaire symétrique lorsque le démodulateur est à sortie ferme, c'est-à-dire lorsqu'il fournit des données discrètes binaires.

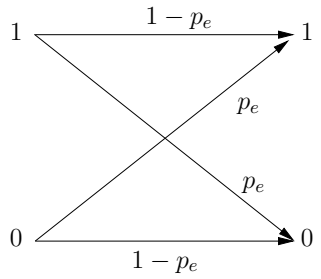


Figure 2.3 — Canal binaire symétrique de probabilité p_e.

2.2.2.2 Canal q-aire symétrique

Un canal q-aire symétrique, appelé aussi QSC ("Q-ary Symmetric Channel"), de probabilité d'erreur p_e est une généralisation du canal BSC. Les entrées $\mathbf{c}$ et les sorties $\mathbf{r}$ appartiennent au corps de Galois GF(q)= $\{0, 1, \alpha, \cdots, \alpha^{q-2}\}$, où α est un élément primitif. Les symboles à l'entrée du canal, c_i, $\forall i \in [\![1, L]\!]$, sont indépendants et uniformément distribués avec une probabilité égale à $1/q$. Un symbole $c_i = a \in$ GF(q) transmis par le canal QSC est reçu erroné avec une probabilité $p_e/(q-1)$ [Mor12]. En d'autres termes, le symbole erroné sera remplacé à la réception par un autre symbole du corps GF(q). La probabilité de recevoir le même symbole transmis est $1 - p_e$. Les probabilités conditionnelles à la sortie du canal sont définies par :

$$\begin{aligned} p(r_i = b|c_i = a) &= \frac{p_e}{q-1} \\ p(r_i = a|c_i = a) &= 1 - p_e \end{aligned} \tag{2.2}$$

avec a et b des symboles distincts du corps GF(q). Nous illustrons dans la figure 2.4 le principe de fonctionnement d'un canal 4-aire symétrique de probabilité d'erreur p_e.

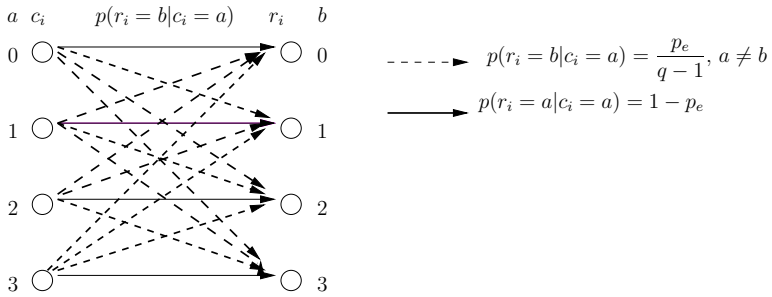


Figure 2.4 — Canal 4-aire symétrique de probabilité p_e

Nous présentons maintenant quelques modèles de canaux plus réalistes.

2.2.2.3 Canal à bruit blanc additif gaussien

Le modèle du canal à Bruit Blanc Additif Gaussien (BBAG) est l'un des plus utilisés dans les simulations d'une chaîne de transmission numérique. En effet, le bruit blanc modélise de manière la plus simple possible l'ensemble des bruits d'origines interne et externe perturbant le signal transmis. Il ajoute aux symboles $\mathbf{c} = \begin{pmatrix} c_1 & \cdots & c_L \end{pmatrix}$ issus d'un modulateur un bruit qui introduit des erreurs, notées $\mathbf{e} = \begin{pmatrix} e_1 & \cdots & e_L \end{pmatrix}$, qui sont générés suivant une distribution gaussienne de moyenne nulle et de variance $\sigma_w^2 = {}^{N_0}/_2$. La sortie du canal $\mathbf{r} = \begin{pmatrix} r_1 & \cdots & r_L \end{pmatrix}$ est donnée par :

$$\mathbf{r} = \mathbf{c} + \mathbf{e} \tag{2.3}$$

La densité de probabilité conditionnelle d'un symbole reçu r_i est définie par :

$$p(r_i|c_i) = \frac{1}{\sqrt{2 \cdot \pi} \cdot \sigma_w} \cdot \exp\left(-\frac{(r_i - c_i)^2}{2 \cdot \sigma_w^2}\right) \tag{2.4}$$

Dans le cadre de cette étude, nous utilisons des modulations d'ordre q pour les codes dans GF(q). Lors d'une transmission des symboles de GF(q) par une modulation d'ordre q, l'ensemble du modulateur, du canal BBAG et du démodulateur est équivalent à un canal non-binaire de probabilité d'erreur qui dépend du type de modulation utilisé [RL09]. Par la suite, nous présentons quelques exemples de modulations d'ordre q lors d'une décision ferme et d'une décision souple. Ces modulations seront utilisées dans nos travaux.

Modulation BPSK

Pour les codes binaires, les bits codés peuvent être modulés par une modulation BPSK (Binary Phase-Shift Keying). Cette modulation convertie les bits en des symboles $\{-1, 1\}$ ($0 \rightarrow -1$ et $1 \rightarrow 1$). Lors d'une transmission de ces symboles sur un canal BBAG, un bruit gaussien de variance $\sigma^2 = \frac{N_0}{2}$ est ajouté. La probabilité d'erreur p_e en tant que expression mathématique indique l'estimation du paramètre TEB. Cette probabilité peut être exprimée par :

$$p_e = \mathcal{Q}\left(\sqrt{2 \cdot R \cdot \frac{E_b}{N_0}}\right) \tag{2.5}$$

où :

$$\mathcal{Q}(x) = \frac{1}{2} \cdot erfc\left(\frac{x}{\sqrt{2}}\right) \tag{2.6}$$

Le paramètre R est définit par le rendement du code. Le calcul de la probabilité p_e est réalisé en sortie du démodulateur lors d'une décision ferme, c'est-à-dire lorsque les symboles $\{-1, 1\}$ sont

convertis en des bits. Lors d'une décision souple, le logarithme du rapport de vraisemblance (LRV) des symboles reçus r_i est calculé en sortie du démodulateur :

$$L(r_i) = \log\left(\frac{Pr(r_i|c_i = 1)}{Pr(r_i|c_i = -1)}\right) = \frac{2 \cdot r_i}{\sigma^2} \tag{2.7}$$

où c_i est le symbole émis. En utilisant le LRV des symboles reçus, le démodulateur est capable de décider si le symbole reçu est 1 ou 0 :

$$\begin{cases} r_i \to 1 \text{ si } L(r_i) > 0 \\ r_i \to 0 \text{ si } L(r_i) < 0 \end{cases} \tag{2.8}$$

La valeur de $L(r_i)$ représente l'information souple et son signe correspond à l'information ferme.

Modulation PAM d'ordre q

On dit une modulation d'ordre q, si sa constellation comporte q états, on la note q-PAM dans le cas de la modulation PAM (Pulse Amplitude Modulation). Comme son nom l'indique, le principe de cette modulation consiste à moduler l'amplitude d'un train d'impulsions. Les symboles obtenus par la modulation PAM d'ordre q appartiennent à l'alphabet $\{(1-q)\cdot d, (3-q)\cdot d, (5-q)\cdot d, \cdots, (q-2)\cdot d, (q-1)\cdot d\}$, où d est une constante. La probabilité d'erreur d'un symbole dans le cas de cette modulation est donnée par :

$$p_e = 2 \cdot \frac{q-1}{q} \cdot \mathcal{Q}\left(\sqrt{\frac{6}{q^2-1} \cdot R \cdot \frac{E_s}{N_0}}\right) \tag{2.9}$$

Modulation QAM d'ordre q

Il s'agit d'une modulation d'amplitude en quadrature, appelée QAM (Quadrature Amplitude Modulation). Avec cette modulation, les symboles de GF(q), c_i, sont convertis en des symboles complexes, notés s_i, qui sont représentés par un couple (a_i, b_i), où a_i est la partie réelle et b_i est la partie imaginaire. Les symboles de ce couple sont indépendants. Ils prennent leur valeurs dans le même alphabet de modulation donnée par : $\{\pm d, \pm 3 \cdot d, \pm 5 \cdot d, \cdots, \pm(q-1) \cdot d\}$ pour $q = 2^{2 \cdot m}$, où d est une constante. Dans ce cas, les symboles a_i et b_i représentent chacun un mot de m bits. Donc, le symbole s_i représente un mot de $2 \cdot m$ bits. Lors d'une transmission, les symboles a_i et b_i sont envoyés sur une voie en phase et une voie en quadrature. Le symbole reçu en entrée du démodulateur est la composition des symboles reçus sur les deux voies. Lors d'une décision ferme, le démodulateur convertit ce symbole en un élément de GF(q). En sortie de ce démodulateur, on peut calculer la probabilité d'erreur d'un symbole, p_e, qui est exprimée par :

$$p_e = 2 \cdot \frac{(\sqrt{q}-1)}{\sqrt{q}} \cdot \mathcal{Q}\left(\sqrt{\frac{3}{q-1} \cdot R \cdot \frac{E_s}{N_0}}\right) \tag{2.10}$$

Cette expression de p_e n'est valable que pour les modulations QAM d'ordre 2^m à un contour carré de constellation, c'est-à-dire lorsque le nombre de bits par symbole, m, est pair. Par contre, pour tout m, la probabilité d'erreur d'un symbole peut être approximée par :

$$p_e \approx 4 \cdot \mathcal{Q}\left(\frac{3}{q-1} \cdot R \cdot \frac{E_s}{N_0}\right) \tag{2.11}$$

Lorsque la décision du démodulateur est souple, pour chaque symbole reçu r_i, un vecteur du logarithme du rapport de vraisemblance de taille $q-1$, noté $\mathbf{L}_{r_i}$, est calculé :

$$\mathbf{L}_{r_i} = \begin{pmatrix} L(c_i = 1) & L(c_i = \alpha) & L(c_i = \alpha^2) & \cdots & L(c_i = \alpha^{q-2}) \end{pmatrix} \tag{2.12}$$

où $\{0, 1, \alpha, \alpha^2, \cdots, \alpha^{q-2}\}$ sont les éléments du corps GF(q), tel que α est un élément primitif, et $L(c_i = \alpha^j)$, pour $j \in [\![0, q-2]\!]$, est définit par :

$$L(c_i = \alpha^j) = \log\left(\frac{Pr(r_i|c_i = \alpha^j)}{Pr(r_i|c_i = 0)}\right) \tag{2.13}$$

Supposons que le mapping utilisé, pour $j \in [\![1, q-2]\!]$, est :

$$\begin{cases} \alpha^j & \Leftrightarrow & s^{(j+1)} \\ \alpha^0 & \Leftrightarrow & s^{(0)} \end{cases}$$

où $s^{(j)}$, pour tout $j \in [\![0, q-1]\!]$ est un symbole de l'alphabet de la modulation. Le LRV $L(c_i = \alpha^j)$ est déterminé par :

$$L(c_i = \alpha^j) = \frac{-1}{N_0} \cdot \left(|r_i - s^{(j+1)}|^2 - |r_i - s^{(0)}|^2\right) \tag{2.14}$$

A partir du vecteur de l'information souples pour le symbole r_i, $\mathbf{L}_{r_i}$, on peut déterminer l'information ferme de ce symbole qui correspond à un symbole décidé du corps de Galois GF(q) donnant le maximum de vraisemblance :

$$r_i = \begin{cases} arg(\max_{\alpha^j \in GF(q)^*}(L(c_i = \alpha^j))) & \text{si le max} > 0 \\ 0 & \text{si le max} \leq 0 \end{cases} \tag{2.15}$$

2.2.2.4 Canal à évanouissements

Les évanouissements sont les atténuations du signal émis lors d'une transmission radio-mobile. Ces évanouissements sont provoqués par les mouvements de l'émetteur et du récepteur, qui entraînent des variations temporelles du canal, et par un environnement de propagation riche en échos, qui se caractérise par de nombreux trajets multiples. Le phénomène de trajets multiples est lié aux interactions électromagnétiques comme la diffraction, la réflexion et la réfraction engendrées par les montagnes, immeubles, voitures, etc...

Le modèle du canal à évanouissements utilise la modélisation par BBAG. Les symboles transmis $\mathbf{c}$ sur ce canal subissent des atténuations d'amplitude et un certain décalage. Le symbole reçu, r_i, à la sortie du canal à évanouissements à L_p trajets multiples s'exprime alors par :

$$r_i = \sum_{l=0}^{L_p-1} a_l \cdot c_{i-\tau_l} + e_i \;\; \forall i \in [\![1, L]\!] \tag{2.16}$$

où e_i représente l'erreur introduite par le bruit BBAG et a_l et τ_l caractérisent respectivement l'atténuation complexe et le retard affectant chaque trajet l. La phase de a_l est en général une variable aléatoire qui suit une distribution uniforme sur $[0, 2 \cdot \pi]$. Par contre, le module de a_l suit une loi de Rayleigh ou de Rice. Le canal multi-trajet peut générer des interférences entre symboles. Afin de compenser les effets des interférences, la technique d'égalisation est souvent appliquée. Plusieurs techniques d'égalisation on été proposées dans les littératures. La technique qui utilise le critère de l'erreur quadratique moyenne, appelée MMSE (Minimum Mean Square Error) présente une complexité raisonnable.

2.2.3 Codage de canal

2.2.3.1 Introduction

Le bloc de codage canal joue un rôle fondamental dans la fiabilité des systèmes de communication. Il est utilisé pour améliorer la qualité des transmissions en protégeant l'information émise contre les erreurs et/ou les perturbations introduites par le canal de transmission. L'idée de base du bloc de codage de canal est d'ajouter des bits ou des symboles de redondance à l'information utile. Les symboles de redondance sont introduits dans les symboles d'information selon une règle de codage. Le taux de codage ou le rendement du code, noté R, est défini par le rapport entre le nombre de symboles d'information en entrée du codeur et le nombre de symboles codés en sortie. Cette notion mesure le degré de redondance dans un codeur. En effet, plus le rendement est faible, plus l'information transmise est redondante et donc protégée. Shannon a énoncé un théorème dans [Sha48] qui fixe une borne maximale pour le rendement R. D'après ce théorème, il est possible de recevoir l'information transmise sans erreur si on choisit un rendement de codage R inférieur à la capacité du canal de transmission. Ce théorème ne donne pas une méthode pour construire de tels codes, mais il fixe une borne théorique pour atteindre une capacité de correction optimale des erreurs. Depuis de nombreuses années, des chercheurs dans le domaine du codage de canal se sont concentrés sur le développement de codes se rapprochant autant que possible de la borne théorique de Shannon.

Afin de détecter et/ou de corriger d'éventuelles erreurs de transmission, le bloc de codage canal fait appel à des codes appelés codes correcteurs d'erreurs. Ces codes ajoutent de la redondance aux symboles d'information qui peut être exploitée par le récepteur à l'aide du bloc de décodage canal pour détecter et/ou corriger les symboles bruités dans le but de retrouver l'information utile. De nos jours, il existe plusieurs méthodes de construction des codes correcteurs d'erreurs qui permettent de réduire le plus possible les erreurs de décodage, tout en assurant un débit de transmission élevé avec une faible complexité du système de codage et de décodage. La plupart des chercheurs se sont limités à des codes correcteurs d'erreurs binaires (GF(2)). Mais, la redécouverte des codes LDPC (Low Density Parity Check) a conduit à étudier les performances des codes non-binaires (dans GF(q), avec $q > 2$) qui ont montré de meilleures capacités de correction que les codes binaires.

Les codes correcteurs d'erreurs ont été classés en deux catégories principales, les codes en bloc et les codes convolutifs. Les codes en bloc sont composés des codes linéaires et des codes non-linéaires. En pratique, seuls les codes en bloc de type linéaire sont utilisés car ils permettent d'utiliser les notions d'algèbre linéaire, ce qui facilite les opérations de codage et de décodage. Ainsi, le formalisme matriciel, où les opérations de calcul sont effectuées dans le corps de Galois GF(q) est utilisé pour représenter les opérations de codage et de décodage. Les codes en bloc linéaires peuvent être regroupés en deux classes : les codes cycliques et les non-cycliques. Le principe d'un code en bloc est de découper les symboles d'information en blocs de taille fixe, notée k. Un bloc d'information va être codé indépendamment des autres blocs pour produire un mot de code de taille n, avec $n > k$. Les premiers codes en bloc linéaires apparus sont les codes de Hamming. Ils ont été développés en 1946 par Richard Wesley Hamming [Ham50]. Ces codes sont capables de détecter trois erreurs et d'en corriger une. Les codes de Hamming sont très utilisés dans les applications liées à l'informatique et aux télécommunications. Depuis l'invention des codes de Reed-Muller [Ree54], il est possible de corriger plus d'une erreur de transmission. Ces codes ont été utilisés de manière spectaculaire lors de la transmission d'images par satellite et sondes spatiales. A partir des codes de Reed-Muller, les codes cycliques de Reed-Solomon [RS60] et de BCH [Hoc59, BRC60] ont été construits. Ces codes utilisent les corps de Galois de cardinal q. Ils représentent les premiers codes non-binaires. Les codes de Reed-Solomon sont un cas particulier des codes BCH. Ils sont largement utilisés pour le stockage et la transmission des données. Les codes LDPC constituent une classe de codes en bloc caractérisés par une matrice de contrôle de parité creuse. Ils ont été développés en 1962, mais ils sont restés longtemps en sommeil à cause de la complexité de leur décodage et de la concurrence

induite par l'apparition des codes de Reed-Solomon qui sont caractérisés par une faible complexité. Grâce à l'amélioration apportée aux techniques de codage et de décodage des codes LDPC, ces codes ont été redécouverts et ont montré des performances s'approchant de la limite de Shannon. Les codes convolutifs développés par Elias en 1954 [Eli54] constituent une deuxième catégorie de codes correcteurs d'erreurs. Ils sont subdivisés en récursifs ou non récursifs. Les turbocodes [Ber06], dans leurs versions d'origine, sont construits en utilisant deux codes convolutifs récursifs. Pour les codes convolutifs, chaque mot de code dépend du mot d'information présent à l'entrée du codeur ainsi que des autres mots d'information précédemment introduits.

Dans ce mémoire, nous nous sommes intéressés aux propriétés des codes correcteurs d'erreurs binaires et non-binaires. Puisque les relations de codage et de décodage pour ces codes sont de type linéaire, un rappel sur le concept de base des codes linéaires q-aires (travaillant dans GF(q), $q \geqslant 2$) est présenté.

2.2.3.2 Codes linéaires q-aires

Nous supposons que la séquence d'information en entrée du codeur est composée de symboles du corps GF(q). Dans le cas des codes en bloc linéaires, cette séquence est découpée en mots d'information de longueur fixe k. Il existe alors q^k mots d'information distincts. Chaque mot d'information est codé en un mot de code de longueur $n > k$. Les mots de code sont composés de symboles du corps GF(q). Puisqu'il existe un seul mot de code par mot d'information, il existe donc q^k mots de code distincts. Un mot de code de longueur n contient $n - k$ symboles de redondance (appelé aussi symboles de parité) ajoutés par l'algorithme de codage.

Définition 2.9. *Un code linéaire, noté $\mathcal{C}$, de longueur n, est un code q-aire si et seulement si les q^k mots du code forment un sous-espace vectoriel de dimension k de l'espace vectoriel de dimension n sur le corps GF(q).*

Si $q = 2$, le code linéaire est dit binaire, sinon il est dit non-binaire. Un mot de code en sortie du codeur est défini comme un vecteur du sous espace vectoriel $\mathcal{C}$, c'est-à-dire du code $\mathcal{C}$. Puisque la dimension de cet espace est k, alors il existe k mots de code qui sont linéairement indépendants et qui forment une base de $\mathcal{C}$ comme espace vectoriel sur GF(q). Ainsi, tous les mots de code sont des combinaisons linéaires des k mots de code d'une telle base.

Définition 2.10. *Une matrice génératrice d'un code linéaire $\mathcal{C}$ de longueur n et de dimension k sur GF(q) en une matrice de taille ($k \times n$) dont les lignes forment une base de $\mathcal{C}$ comme GF(q)-espace vectoriel.*

Tous les mots de code sont générés par des combinaisons linéaires des lignes d'une matrice génératrice, notée $\mathbf{G}$.

Exemple 2.10.

Considérons un code linéaire 2-aire (c'est-à-dire dans GF(2)) de paramètres $k = 2$ et $n = 3$, noté $\mathcal{C}(3, 2)$.

Il existe $2^k = 4$ mots d'information possibles qui peuvent être codés par ce code pour obtenir 4 mots de code différents. Le tableau 2.1 présente ces 4 mots d'information et leurs mots de code associé obtenus par ajout d'un bit de parité pour un code $\mathcal{C}(3, 2)$ particulier.

Mots d'information	Mots de code
$\begin{pmatrix}0 & 0\end{pmatrix}$	$\begin{pmatrix}0 & 0 & 0\end{pmatrix}$
$\begin{pmatrix}0 & 1\end{pmatrix}$	$\begin{pmatrix}0 & 1 & 1\end{pmatrix}$
$\begin{pmatrix}1 & 0\end{pmatrix}$	$\begin{pmatrix}1 & 0 & 1\end{pmatrix}$
$\begin{pmatrix}1 & 1\end{pmatrix}$	$\begin{pmatrix}1 & 1 & 0\end{pmatrix}$

Tableau 2.1 — Les différents mots d'information et mots de code

Soient $\mathbf{c}_1 = \begin{pmatrix}1 & 0 & 1\end{pmatrix}$ et $\mathbf{c}_2 = \begin{pmatrix}0 & 1 & 1\end{pmatrix}$ deux mots de code qui forment une base de $\mathcal{C}$. Alors la matrice suivante $\mathbf{G}$ est une matrice génératrice de $\mathcal{C}$:

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

Définition 2.11. *Soit* $\mathbf{u}$ *un vecteur de symboles d'information de longueur* k*. Un mot de code, noté* $\mathbf{c}$*, de* $\mathcal{C}$ *de longueur* n *est obtenu par :*

$$\mathbf{c} = \mathbf{u} \cdot \mathbf{G} \tag{2.17}$$

L'équation (2.17) correspond à l'opération de codage. Un code linéaire $\mathcal{C}$ possède en général plusieurs matrices génératrices puisqu'un espace vectoriel peut être défini par plusieurs bases. Donc, chaque choix de base nous donne une matrice génératrice. Le rang de $\mathbf{G}$ est égal à la dimension du code k puisque les lignes sont linéairement indépendantes.

Exemple 2.11.

Considérons par exemple le code de Hamming de paramètres $n = 7$ et $k = 4$ qui est défini par une matrice génératrice $\mathbf{G}$ sous la forme :

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

Soit $\mathbf{u} = \begin{pmatrix}1 & 0 & 1 & 0\end{pmatrix}$ un mot d'information. En utilisant l'équation (2.17), le résultat de codage de $\mathbf{u}$ donne le mot de code $\mathbf{c}$ tel que :

$$\begin{aligned}\mathbf{c} &= \begin{pmatrix}1 & 0 & 1 & 0\end{pmatrix} \cdot \begin{pmatrix}1 & 1 & 0 & 1 & 0 & 0 & 0\\ 0 & 1 & 1 & 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 1 & 0 & 1 & 0\\ 0 & 0 & 0 & 1 & 1 & 0 & 1\end{pmatrix}\\ &= \begin{pmatrix}1 & 1 & 1 & 0 & 0 & 1 & 0\end{pmatrix}\end{aligned}$$

Définition 2.12. *Deux codes* $\mathcal{C}$ *et* $\mathcal{C}'$ *caractérisés respectivement par les matrices génératrices* $\mathbf{G}$ *et* $\mathbf{G}'$ *sont équivalents s'il existe une matrice inversible de taille* $(k \times k)$*, notée* $\mathbf{A}$*, et une matrice de permutation de taille* $(n \times n)$*, notée* $\mathbf{B}$*, vérifiant :*

$$\mathbf{G} = \mathbf{A} \cdot \mathbf{G}' \cdot \mathbf{B} \tag{2.18}$$

Exemple 2.12.

Suite de l'exemple 2.11.

On peut obtenir une autre matrice génératrice $\mathbf{G}'$ à partir de $\mathbf{G}$ en remplaçant la première

ligne de $\mathbf{G}$ par la somme des deux première lignes. Alors, la matrice $\mathbf{G}'$ sera :

$$\mathbf{G}' = \begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix} = \mathbf{A} \cdot \mathbf{G} \cdot \mathbf{B}$$

avec :

$$\mathbf{A} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \text{ et } \mathbf{B} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} = \mathbf{I}_8$$

En faisant le codage de $\mathbf{u} = \begin{pmatrix} 1 & 0 & 1 & 0 \end{pmatrix}$ par $\mathbf{G}'$, on obtient un autre mot de code $\mathbf{c}' = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}$ qui est différent de $\mathbf{c}$.

Parmi les codes équivalents à un code linéaire, il existe toujours un code équivalent systématique tel que sa matrice génératrice $\mathbf{G}$ est sous la forme :

$$\mathbf{G} = \begin{pmatrix} \mathbf{P} & \mathbf{I}_k \end{pmatrix} \tag{2.19}$$

où $\mathbf{I}_k$ est la matrice identité de taille $(k \times k)$ et $\mathbf{P}$ est une $(k \times (n-k))$ matrice. Dans ce cas, en utilisant l'équation (2.17), un mot de code $\mathbf{c}$ d'un codeur systématique est obtenu par : $\mathbf{c} = \begin{pmatrix} \mathbf{u} \cdot \mathbf{P} & \mathbf{u} \end{pmatrix}$. Ainsi, les $n-k$ premiers symboles dans $\mathbf{c}$ sont les symboles de redondance ou de parité et les k symboles restants sont les symboles d'information.

Exemple 2.13.

Suite de l'exemple 2.11.

La matrice $\mathbf{G}$ est équivalente à une autre matrice génératrice du code $\mathcal{C}$, notée $\mathbf{G}''$, qui est sous une forme systématique :

$$\mathbf{G}'' = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} = \mathbf{A}' \cdot \mathbf{G} \cdot \mathbf{B}'$$

avec :

$$\mathbf{A}' = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \text{ et } \mathbf{B}' = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Alors, le codage de $\mathbf{u} = \begin{pmatrix} 1 & 0 & 1 & 0 \end{pmatrix}$ par $\mathbf{G}''$ donne le mot de code $\mathbf{c}'' = \begin{pmatrix} 0 & 0 & 1 & 1 & 0 & 1 & 0 \end{pmatrix}$.

Définition 2.13. *Le code dual d'un code linéaire de dimension k et de longueur n est le code $\mathcal{C}^\perp$ défini par :*

$$\mathcal{C}^\perp = \left\{ \mathbf{h} \in GF(q)^n \ : \ \mathbf{c} \cdot \mathbf{h}^T = 0, \forall \mathbf{c} \in \mathcal{C} \right\} \tag{2.20}$$

Le code dual $\mathcal{C}^\perp$ est défini par une base formée par $n-k$ mots de code de longueur n qui sont

linéairement indépendants. Donc, un mot de code de $\mathcal{C}^{\perp}$ est une combinaison linéaire de ces $n-k$ mots de code de la base. Ainsi, le code linéaire $\mathcal{C}^{\perp}$ est engendré par une $((n-k) \times n)$ matrice appelée matrice de contrôle de parité, notée $\mathbf{H}$, qui vérifie :

$$\mathbf{G} \cdot \mathbf{H}^T = \mathbf{0} \tag{2.21}$$

Une ligne de la matrice $\mathbf{H}$ est appelée une équation ou relation de parité. Donc, un code $\mathcal{C}$ de longueur n est caractérisé par $n-k$ équations de parité. Si $\mathbf{G}$ est sous une forme systématique alors une matrice de contrôle de parité $\mathbf{H}$ peut être choisie sous la forme $\mathbf{H} = \begin{pmatrix}\mathbf{I}_{n-k} & -\mathbf{P}^T\end{pmatrix}$, où $\mathbf{I}_{n-k}$ est la matrice identité de taille $((n-k) \times (n-k))$. D'après la relation (2.21), on peut déduire qu'un mot $\mathbf{c}$ appartient au code $\mathcal{C}$ si et seulement s'il vérifie la relation $\mathbf{c} \cdot \mathbf{h}^T = 0$ pour tout $\mathbf{h} \in \mathcal{C}^{\perp}$. En général, les opérations de décodage sont basées sur l'utilisation de la matrice de contrôle de parité $\mathbf{H}$ afin de détecter la présence d'erreurs dans les mots de code reçus.

Exemple 2.14.

Suite de l'exemple 2.13.

Une matrice de contrôle de parité du code précédent peut s'écrire :

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

Nous pouvons vérifier que la matrice génératrice $\mathbf{G}''$ de l'exemple 2.13 qui est sous forme systématique vérifie l'équation (2.21). Bien qu'elle ne soit pas sous la forme systématique, la matrice $\mathbf{G}$ de l'exemple 2.11 vérifie également la relation (2.21) car elle génère le même code.

Les paramètres n et k sont indispensables pour caractériser les codes linéaires, mais il existe un autre paramètre important qui est la distance de Hamming du code.

Définition 2.14. *Soient* $\mathbf{c}$ *et* $\mathbf{c}'$ *deux mots de code. La distance de Hamming, noté* $d_H(\mathbf{c}, \mathbf{c}')$*, est le nombre de symboles différents entre* $\mathbf{c}$ *et* $\mathbf{c}'$*.*

Pour un code $\mathcal{C}$, chaque mot de code a un poids appelé poids de Hamming.

Définition 2.15. *Soit* $\mathbf{c}$ *un mot de code. Le poids de Hamming de* $\mathbf{c}$*, noté* $w_H(\mathbf{c})$*, est le nombre de symboles non nuls.*

Ainsi, la distance de Hamming entre deux mots de code $\mathbf{c}$ et $\mathbf{c}'$ d'un code linéaire peut s'écrire :

$$d_H(\mathbf{c}, \mathbf{c}') = w_H(\mathbf{c} - \mathbf{c}') \tag{2.22}$$

Exemple 2.15.

Soient $\mathbf{c} = \begin{pmatrix}\alpha & 0 & 0 & 1 & \alpha^2 & \alpha^4 & 0\end{pmatrix} = \begin{pmatrix}2 & 0 & 0 & 1 & 4 & 6 & 0\end{pmatrix}$ et $\mathbf{c}' = \begin{pmatrix}\alpha^5 & \alpha^3 & 0 & 0 & 1 & \alpha^6 & \alpha^4\end{pmatrix} = \begin{pmatrix}7 & 3 & 0 & 0 & 1 & 5 & 6\end{pmatrix}$ deux mots de code dans GF(2^3), où α est une racine du polynôme primitif $x^3 + x + 1$. Les poids de Hamming de $\mathbf{c}$ et $\mathbf{c}'$ sont $w_H(\mathbf{c}) = 4$ et $w_H(\mathbf{c}') = 5$. La différence de ces deux mots de code $\mathbf{c} - \mathbf{c}' = \begin{pmatrix}\alpha^6 & \alpha^3 & 0 & 1 & \alpha^6 & \alpha^3 & \alpha^4\end{pmatrix} = \begin{pmatrix}5 & 3 & 0 & 1 & 5 & 3 & 6\end{pmatrix}$ a pour poids de Hamming 6, ce qui donne une distance de Hamming $d_H(\mathbf{c}, \mathbf{c}') = 6$. Le lecteur intéressé pourra se référer à l'annexe B où un tableau d'addition dans GF(2^3) est présenté. Rappelons que l'addition et la soustraction dans GF(2^m) sont équivalentes.

Pour mesurer la capacité de détection et de correction d'un code $\mathcal{C}$, la notion de distance minimale peut être utilisée.

Définition 2.16. *La distance minimale d'un code linéaire $\mathcal{C}$ est le poids de Hamming minimal des mots de code non nuls de $\mathcal{C}$:*

$$d_{min} = \min_{\mathbf{c} \neq \mathbf{0}} w_H(\mathbf{c}) \quad \forall \mathbf{c} \in \mathcal{C} \tag{2.23}$$

La distance minimale peut être définie à partir de la matrice de contrôle de parité $\mathbf{H}$. Elle correspond au nombre minimal des colonnes de $\mathbf{H}$ linéairement dépendantes.

Théorème 2.9. *Soit $\mathbf{c}$ un mot transmis appartenant à un code $\mathcal{C}$ de distance minimale d_{min} et soit $\mathbf{r}$ un mot reçu comportant t erreurs.*

- *Si $t < d_{min}$, le code $\mathcal{C}$ est capable de détecter les erreurs dans $\mathbf{r}$.*
- *Si $t < d_{min}/2$, le code $\mathcal{C}$ est capable de corriger les erreurs dans $\mathbf{r}$.*

Par conséquent, un code $\mathcal{C}$ de distance minimale d_{min} permet de détecter jusqu'à $d_{min} - 1$ erreurs et de corriger jusqu'à $\lfloor (d_{min} - 1)/2 \rfloor$ erreurs.

Exemple 2.16.

Suite de l'exemple 2.13.

Pour le code de Hamming de paramètres $n = 7$ et $k = 4$, il existe $2^k = 16$ mots de code possibles. Ces mots de code et leurs poids sont présentés dans le tableau 2.2.

Mots de code c	$w_H(\mathbf{c})$
$(0\ 0\ 0\ 0\ 0\ 0\ 0)$	0
$(1\ 0\ 0\ 0\ 0\ 1\ 1)$	3
$(0\ 1\ 0\ 0\ 1\ 0\ 1)$	3
$(0\ 0\ 1\ 0\ 1\ 1\ 0)$	3
$(0\ 0\ 0\ 1\ 1\ 1\ 1)$	4
$(1\ 1\ 0\ 0\ 1\ 1\ 0)$	4
$(1\ 0\ 1\ 0\ 1\ 0\ 1)$	4
$(1\ 0\ 0\ 1\ 1\ 0\ 0)$	3
$(0\ 1\ 1\ 0\ 0\ 1\ 1)$	4
$(0\ 1\ 0\ 1\ 0\ 1\ 0)$	3
$(0\ 0\ 1\ 1\ 0\ 0\ 1)$	3
$(1\ 1\ 1\ 0\ 0\ 0\ 0)$	3
$(1\ 1\ 0\ 1\ 0\ 0\ 1)$	4
$(1\ 0\ 1\ 1\ 0\ 1\ 0)$	4
$(0\ 1\ 1\ 1\ 1\ 0\ 0)$	4
$(1\ 1\ 1\ 1\ 1\ 1\ 1)$	7

Tableau 2.2 — Les différents mots de code $\mathbf{c}$ et leurs poids $w_H(\mathbf{c})$

Nous pouvons voir que le poids minimal des mots de code non-nuls présentés dans ce tableau est égal à 3. D'après l'équation (2.23), la distance minimale de ce code est $d_{min} = 3$.

Une autre méthode permettant la détermination de d_{min} est basée sur l'utilisation de la matrice de contrôle de parité $\mathbf{H}$ de l'exemple 2.14. Notons $\mathbf{v}_i$, pour tout $i \in [\![1, n = 7]\!]$, la i-ème colonne de $\mathbf{H}$. les différents ensembles de colonnes linéairement dépendantes de $\mathbf{H}$ dont le nombre est minimal sont présentés dans le tableau 2.3.

Colonnes dépendantes dans H	Nombre de colonnes
$\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_4\}$	3
$\{\mathbf{v}_1, \mathbf{v}_3, \mathbf{v}_7\}$	3
$\{\mathbf{v}_1, \mathbf{v}_5, \mathbf{v}_6\}$	3
$\{\mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_5\}$	3
$\{\mathbf{v}_2, \mathbf{v}_6, \mathbf{v}_7\}$	3
$\{\mathbf{v}_3, \mathbf{v}_4, \mathbf{v}_6\}$	3
$\{\mathbf{v}_4, \mathbf{v}_5, \mathbf{v}_6\}$	3

Tableau 2.3 — Les différents ensemble de colonnes dépendantes dans **H**

D'après ce tableau, on peut voir que le nombre minimum de colonnes dépendantes dans **H** est 3, ce qui permet d'en déduire la distance minimale $d_{min} = 3$. Ce résultat confirme celui trouvé par la première méthode.

Ainsi, ce code sera capable de détecter 2 erreurs au moins dans un mot de code reçu et d'en corriger une seule.

2.3 Codes correcteurs d'erreurs abordés dans cette étude

En raison de la complexité du processus de codage, mais surtout celle du décodage, la plupart des recherches sur les codes correcteurs d'erreurs se sont limitées à des données binaires, c'est-à-dire des éléments de GF(2). Les codes correcteurs d'erreurs non-binaires les plus utilisés jusqu'à peu étaient les codes de Reed-Solomon. Récemment, des algorithmes de décodage à faible complexité pour les codes LDPC non-binaires ont vu le jour [DM98b]. Il en va de même pour les turbocodes non-binaires, qui ont de bonnes propriétés comme codeurs externes pour le codage de symboles non-binaires correspondant à des modulations numériques à grand nombre d'états, suscitant ainsi un grand intérêt [BS08].

Afin d'appliquer nos algorithmes d'identification aveugle des paramètres du codeur, notre étude se concentre sur les codes correcteurs d'erreurs binaires ($q = 2$) et non-binaires ($q > 2$). Dans ce mémoire, nous présentons uniquement les codes que nous avons abordés dans nos travaux d'identification. Pour ces codes, nous présentons leurs propriétés et leurs principes de codage sans introduire les algorithmes de décodage dans ce chapitre. Pour la famille des codes en bloc, nous considérons les codes LDPC sous forme binaire et non-binaire et les codes Reed-Solomon. Pour la famille des codes convolutifs, les propriétés des codes convolutifs binaires et non-binaires seront exposées.

2.3.1 Codes en blocs

2.3.1.1 Codes LDPC binaires

Les codes LDPC binaires sont des codes correcteurs d'erreurs en bloc linéaires permettant de s'approcher de la limite de Shannon. Ils ont été inventés par Gallager [Gal63, Gal62] mais ignorés pendant trente années dues à la complexité du décodage. Dans les années 1990s, ces code ont été redécouverts et généralisés par Mackay et al. [MN96, Mac99] qui ont remis en lumière l'intérêt des codes LDPC à l'aide de la nouvelle représentation des codes basée sur la graphe bipartite de Tanner [Tan81].

Puisque les codes LDPC binaires représentent une classe des codes linéaires q-aires avec $q = 2$, ils vérifient aussi les propriétés des codes linéaires présentées dans la section 2.2.3.2. La différence entre les codes LDPC et les autres codes est la caractéristique de la matrice de contrôle de parité **H** qui est toujours creuse pour les codes LDPC, c'est-à-dire qu'elle comporte beaucoup de '0' et un très faible nombre d'éléments non nuls. Ce type de matrice permet de simplifier le schéma de

décodage et donc d'avoir un décodage rapide.

La matrice **H** des codes LDPC peut être représentée par un graphe bipartite, appelé graphe de Tanner. Ce graphe représente graphiquement les relations entre les n bits d'un mot de code et les $n-k$ équations de parité. Les nœuds de variable, notés $c_i, \forall i \in \{1, ..., n\}$, symbolisés par des cercles sont associés aux coordonnées des mots de code. Les nœuds de parité, notés p_j, $\forall j \in [\![1, n-k]\!]$, symbolisés par des rectangles sont associés aux équations de parité. Un bit non nul dans la j-ème ligne et la i-ème colonne de la matrice **H** crée une connexion dans le graphe entre le nœud de variable c_i et le nœud de parité p_j, c'est-à-dire les arrêtes du graphe correspondent aux coefficients non-nuls de la matrice de parité. Pour un nœud de parité p_j, la somme des arrêtes qui relient les nœuds de variable et ce nœud doit être nulle, afin de vérifier la relation (2.20).

Lorsque le nombre de bits non nuls dans chaque ligne et chaque colonne de **H** est constant, on parle alors de code LDPC régulier. Notons w_c le poids des colonnes et w_l le poids des lignes. Chaque nœud de variable est connecté à w_c nœuds de parité et chaque nœud de parité est connecté à w_l nœuds de variable. Pour ces codes, il existe une relation entre le rendement du code $R = k/n$ et les poids w_c et w_l. Cette relation est définie par :

$$w_c = w_l \cdot (1 - R) \tag{2.24}$$

Si toutes les colonnes et toutes les lignes n'ont pas le même poids, le code LDPC est dit irrégulier.

Exemple 2.17.

Prenons l'exemple d'un code LDPC de paramètres $n = 6$, $k = 3$ et de matrice de contrôle de parité :

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

Notre objectif est de générer le graphe de Tanner de ce code. En observant sa matrice **H**, on peut remarquer que le graphe doit contenir 6 nœuds de variables $\{c_1, c_2, c_3, c_4, c_5, c_6\}$ et 3 nœuds de parité $\{p_1, p_2, p_3\}$. En regardant la première colonne de **H**, les bits à 1 se trouvent à la deuxième et à la troisième ligne. Donc, le nœud de variable c_1 est connecté aux nœuds de parité p_2 et p_3. Par analogie, les autres nœuds sont connectés de la même façon. Le graphe de Tanner généré par cette matrice **H** est représenté dans la figure 2.5.

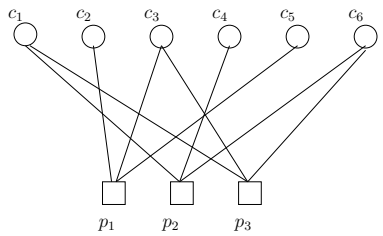


Figure 2.5 — Graphe de Tanner d'un code LDPC binaire de rendement $^3/_6$

Pour chaque nœud de parité p_j, pour $j \in \{1, 2, 3\}$, les nœuds de variable d'un mot de code vérifient :

- $p_1 : c_2 + c_3 + c_5 = 0$,
- $p_2 : c_1 + c_4 + c_6 = 0$,
- $p_1 : c_1 + c_3 + c_6 = 0$

Notons que le code LDPC présenté dans cet exemple est irrégulier puisque toutes les colonnes n'ont pas le même poids.

Le graphe de Tanner est usuellement utilisé comme support pour décrire les algorithmes de décodage des codes LDPC. En effet, un algorithme de décodage itératif est appliqué aux nœuds de variable du graphe. Cet algorithme est appelé algorithme de propagation de croyance où les deux types de nœuds (nœuds de variable et de parité) échangent les informations via les branches du graphe. Afin d'appliquer un décodage performant sur les codes LDPC, il est nécessaire de vérifier qu'il n'existe aucun cycle court en particulier de longueur 4 ou 6 dans le graphe de Tanner. L'existence de ces cycles mène à l'apparition de planchers d'erreurs qui se manifestent par la dégradation soudaine de la pente du taux d'erreur dans la zone de fort rapport signal à bruit, ce qui provoque l'échec de l'opération de décodage [Ric03]. Ces problèmes de cycle et de plancher d'erreurs ont mené les auteurs dans [KLF01] à développer une méthode de construction des codes LDPC basée sur les géométries finies qui permet d'obtenir des bas plancher d'erreurs.

Construire un code LDPC revient à définir une matrice de contrôle de parité qui vérifie certaines conditions. Dans la littérature, il existe différentes méthodes de construction. Elles peuvent être classées en deux catégories principales : les méthodes algébriques et les méthodes aléatoires basées sur l'utilisation de l'ordinateur. Les méthodes algébriques sont basées sur des techniques de géométries finies [KLF01, XCD+07] ou sur des techniques combinatoires [Vas02, JW01, Vas01]. Ces techniques sont caractérisées par la rapidité de codage grâce à la simplicité de la structure de $\mathbf{H}$. Parmi les méthodes aléatoires, il existe la construction de Gallager [Gal62], la construction de Mackay [MN96] et la méthode d'évolution de densité [RSU01]. Ces méthodes offrent de bonnes performances s'approchant de la limite de Shannon pour des codes de grande taille. Cependant, leur inconvénient est la complexité de décodage due à l'irrégularité du poids des colonnes de la matrice $\mathbf{H}$. Une meilleure méthode de construction serait une méthode qui offrirait de bonnes performances s'approchant de la limite de Shannon avec une faible complexité de décodage. Le développement d'une telle méthode a suscité un intérêt considérable chez de nombreux chercheurs.

En choisissant l'une des méthodes de construction citées, on peut définir une matrice de contrôle de parité d'un code LDPC binaire. Puisque les codes LDPC constituent une classe des codes linéaires, alors le codage peut être réalisé à l'aide de la matrice génératrice $\mathbf{G}$ en utilisant l'équation (2.17). Pour construire la matrice $\mathbf{G}$, on peut utiliser l'équation (2.21) qui nous permet de déduire la matrice équivalente de $\mathbf{G}$ sous la forme systématique à partir de la matrice $\mathbf{H}$. Ensuite, d'après la définition 2.12, la matrice $\mathbf{G}$ peut être déterminée en utilisant une matrice équivalente.

2.3.1.2 Codes LDPC non-binaires

Les codes LDPC construits sur des corps de Galois non-binaires ont été initiés par Davey et Mackay dans [DM98b]. Dans leur article, ils ont généralisé l'algorithme de décodage itératif de propagation de croyance des codes LDPC binaires au cas des codes LDPC dans GF(q). Ils ont montré que les codes LDPC non-binaires ont de meilleurs performances que les codes LDPC binaires en termes de taux d'erreurs binaires. Ces codes ont montré en particulier de meilleurs performances pour des dimensions du corps de Galois $q < 64$ [VDV+10], pour des modulations à grand nombre d'états [SF02, DCG04] et pour des codes de petites tailles [MD99]. Cependant, l'inconvénient de ces codes est leur complexité de décodage pour des tailles de corps de Galois plus grandes. Le développement d'algorithmes simplifiés pour le décodage des codes LDPC non-binaires fait l'objet des travaux de plusieurs chercheurs [BD03, CDE+05, DF05, VDV+10, ZC11].

Les codes LDPC non-binaires sont également une classe des codes linéaires q-aires qui utilisent la matrice génératrice $\mathbf{G}$ pour le codage et la matrice de contrôle de parité $\mathbf{H}$ pour le décodage. Ces matrices sont définis dans le corps de Galois GF(q)= $\{0, 1, \alpha, \alpha^2, \cdots, \alpha^{q-2}\}$, où α est un élément primitif. Ainsi, les codes LDPC binaires représentent un cas particulier de codes LDPC dans GF(q)

en considérant $q = 2$. Comme les codes LDPC binaires, les codes LDPC non-binaires sont définis par une matrice de contrôle de parité creuse $\mathbf{H}$ de taille $((n-k) \times n)$ avec des coefficients, notés $h_{j,i}$, $\forall j \in [\![1, n-k]\!]$ et $\forall i \in [\![1, n]\!]$, éléments du corps GF($q$). Les mots de code sont des vecteurs de taille n ayant comme éléments c_i des symboles de GF(q). Un mot appartenant au code si et seulement s'il vérifie les relations de parité qui s'écrivent :

$$\sum_{i=1}^{n} h_{j,i} \cdot c_i = 0, \ \forall j \in [\![1, n-k]\!] \tag{2.25}$$

Les opérations d'addition et de multiplication sont effectuées dans le corps GF(q).

La matrice $\mathbf{H}$ des codes LDPC non-binaires peut être représentée par le graphe de Tanner comme décrit dans la section 2.3.1.1. Cependant, un troisième type de nœud, appelé nœud de fonction, sera ajouté à ce graphe pour les coefficients $h_{j,i} \neq 0$ [DF07, Sha10]. Ce nœud correspond à la multiplication du nœud de variable c_i par le coefficient non nul $h_{j,i}$. Un nœud de variable c_i et un nœud de parité p_j sont reliés par l'intermédiaire d'un nœud de fonction lorsque le coefficient $h_{j,i} \in$GF(q) est non nul.

Pour construire un code LDPC non-binaire, on construit tout d'abord la matrice de contrôle de parité du code LDPC binaire, puis on remplace les bits à 1 dans $\mathbf{H}$ par des symboles non nuls du corps de Galois GF(q). Ces symboles peuvent être choisis aléatoirement comme expliqué dans [HEA05].

Exemple 2.18.

Reprenons le même exemple du code LDPC($n = 6$, $k = 3$). Pour construire ce code dans GF(2^3), on utilise la matrice de contrôle de parité du code LDPC binaire présentée dans l'exemple 2.17. Les bits à 1 de la matrice de parité sont remplacés par des symboles non nuls du corps GF(2^3)=$\{0, 1, \alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6\}$. La matrice $\mathbf{H}$ obtenue est :

$$\mathbf{H} = \begin{pmatrix} 0 & \alpha^6 & \alpha & 0 & \alpha^6 & 0 \\ \alpha^4 & 0 & 0 & \alpha^2 & 0 & \alpha^3 \\ \alpha^3 & 0 & 1 & 0 & 0 & \alpha^5 \end{pmatrix}$$

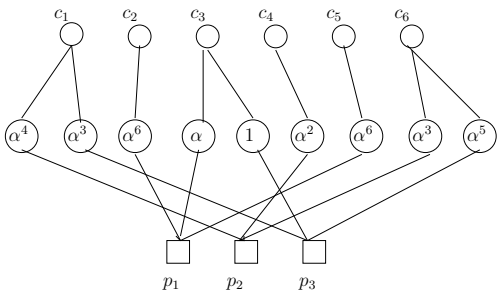


Figure 2.6 — Graphe de Tanner d'un code LDPC(6, 3) dans GF(2^3)

Le graphe de Tanner décrivant la matrice $\mathbf{H}$ est représenté sur la figure 2.6. Pour chaque nœud de parité p_j, la somme des nœuds de variable connectés à ce nœud multiplié par les nœuds de fonction est nulle :

- $p_1 : c_2 \cdot \alpha^6 + c_3 \cdot \alpha + c_5 \cdot \alpha^6 = 0$,
- $p_2 : c_1 \cdot \alpha^4 + c_4 \cdot \alpha^2 + c_6 \cdot \alpha^3 = 0$,

- $p_1 : c_1 \cdot \alpha^3 + c_3 + c_6 \cdot \alpha^5 = 0$

2.3.1.3 Codes de Reed-Solomon

Les codes de Reed-Solomon font partie des codes en bloc linéaires cycliques.

Définition 2.17. *Un code linéaire est cyclique lorsque la permutation à droite de tout un mot de code donne un mot de code.*

Soit $\mathbf{c} = \begin{pmatrix} c_1 & c_2 & \cdots & c_n \end{pmatrix}$ un mot de code qui appartient à un code cyclique $\mathcal{C}$ de longueur n. Alors, le mot $\mathbf{c}' = \begin{pmatrix} c_n & c_1 & c_2 & \cdots & c_{n-1} \end{pmatrix}$ ou le mot $\mathbf{c}'' = \begin{pmatrix} c_{n-1} & c_n & c_1 & c_2 & \cdots & c_{n-1} \end{pmatrix}$ appartient également à $\mathcal{C}$.

Exemple 2.19.

Considérons un code en bloc linéaire cyclique de paramètres $k = 2$ et $n = 3$ qui inclut le mot de code $\begin{pmatrix} 1 & 1 & 0 \end{pmatrix}$. Ce code peut aussi générer le mot de code $\begin{pmatrix} 0 & 1 & 1 \end{pmatrix}$ par permutation à droite ou le mot de code $\begin{pmatrix} 1 & 0 & 1 \end{pmatrix}$ par permutation à gauche.

En utilisant les représentations polynomiales, toute permutation circulaire (à droite ou à gauche) pour un mot de code $\mathbf{c}$ peut s'exprimer en multipliant le polynôme $c(x) = c_1 + c_2 \cdot x + \cdots + c_n \cdot x^{n-1} \in \mathrm{GF}(q)[x]/(x^n - 1)$, qui représente le mot de code $\mathbf{c}$, par un polynôme $p(x)$ modulo $x^n - 1$. En effet, la permutation à droite pour obtenir le polynôme $c'(x)$ de $\mathbf{c}'$ est effectuée par : $x \cdot c(x) \bmod (x^n - 1) = c_n + c_1 \cdot x + c_2 \cdot x^2 + \cdots + c_{n-1} \cdot x^{n-1} = c'(x)$. Par conséquent, lorsqu'un mot de code d'un code cyclique $\mathcal{C}$ où $\mathbf{c}$ est inclut est obtenu après p permutations, sa représentation polynomiale s'écrit : $x^p \cdot c(x) \bmod (x^n - 1)$ avec $p \in \mathbb{N}$.

La théorie des codes cycliques permet de montrer que tout mot de code d'un code cyclique est un multiple d'un polynôme $g(x)$, appelé polynôme générateur. Son degré est $n - k$:

$$c(x) = u(x) \cdot g(x) \tag{2.26}$$

où $u(x)$ est la représentation polynomiale dans la base canonique de la séquence d'information $\mathbf{u}$ de degré inférieur ou égale à k. Un code cyclique $\mathcal{C}(n, k)$ possède un unique polynôme générateur $g(x)$ diviseur de $x^n - 1$.

Exemple 2.20.

Le polynôme $x^7 - 1$ est décomposé en facteurs irréductibles dans GF(2)[x] :

$$x^7 - 1 = (x + 1) \cdot (x^3 + x^2 + 1) \cdot (x^3 + x + 1) \tag{2.27}$$

Nous remarquons que les polynômes irréductibles qui divisent $x^7 - 1$ sont de degrés 1, 3 et 3. Le produit de toute combinaison de ces polynômes permet de construire un polynôme générateur pour un code cyclique $\mathcal{C}(7, k)$. Le paramètre k dépend du polynôme choisi. En effet, si on prend le polynôme $g(x)$ de degré 6 :

$$g(x) = (x^3 + x^2 + 1) \cdot (x^3 + x + 1)$$

le code cyclique en résulte est de dimension $k = 7 - 6 = 1$. Avec le polynôme générateur $g(x) = (x + 1) \cdot (x^3 + x^2 + 1)$, la dimension du code est alors $k = 7 - 4 = 3$.

Le code de Reed-Solomon est cyclique, il est donc défini par un polynôme générateur $g(x)$ de degré $n - k$. Ce polynôme possède $2 \cdot t$ racines qui sont des puissances de l'élément primitif α de

GF(q). Il peut s'écrire :

$$g(x) = (x - \alpha) \cdot (x - \alpha^2) \cdots (x - \alpha^{2 \cdot t}) \tag{2.28}$$

Un code de Reed-Solomon de paramètres n et k est noté RS(n, k). C'est un code cyclique non-binaire construit dans le corps GF(q) qui vérifie les propriétés suivantes :

- le nombre de symboles dans un mot de code est $n = q - 1$,
- la capacité de correction des erreurs dans un mot de code, notée t, dépend des paramètres n et k tel que : $2 \cdot t = n - k$.

Exemple 2.21.

Prenons l'exemple du code RS$(7, 3)$ dans GF(2^3) de longueur $n = 7$ et de capacité de correction $t = 2$ où les éléments du corps de Galois sont construits à l'aide du polynôme primitif $x^3 + x + 1$. Soit α une racine de ce polynôme primitif. En utilisant l'expression (2.28), le polynôme générateur de degré $2 \cdot t = 4$ peut s'écrire :

$$\begin{aligned} g(x) &= (x - \alpha) \cdot (x - \alpha^2) \cdot (x - \alpha^3) \cdot (x - \alpha^4) \\ &= x^4 + \alpha^3 \cdot x^3 + x^2 + \alpha \cdot x + \alpha^3 \end{aligned}$$

Le lecteur intéressé pourra trouver en annexe B le tableau d'addition des éléments du corps GF(2^3) écrits à l'aide du polynôme primitif $x^3 + x + 1$.

Notons que les coefficients de $g(x)$ sont des éléments du GF(2^3). Ce polynôme permet de construire le code RS$(7, 3)$ dans GF(2^3).

Pour effectuer le codage du message $\mathbf{u}$ avec les codes RS, on peut utiliser l'opération de codage des codes cycliques donnée par l'équation (2.26). Puisque les codes RS sont non-binaires, la complexité de cette opération devient plus importante pour de grandes dimensions de corps de Galois GF(q). Une manière plus simple et à faible complexité est d'utiliser le codage systématique en se basant sur les propriétés des codes cycliques [Tod05]. De ce fait, le polynôme $u(x)$ qui représente la séquence d'information $\mathbf{u}$ est tout d'abord multiplié par le polynôme $x^{2 \cdot t}$, puis on le divise par le polynôme générateur $g(x)$, alors on obtient :

$$x^{2 \cdot t} \cdot u(x) = Q(x) \cdot g(x) + P(x) \tag{2.29}$$

tels que les polynômes $Q(x)$ et $P(x)$ correspondent respectivement au quotient et au reste de la division de $x^{2 \cdot t}$ par $g(x)$. Ainsi, la représentation polynomiale $c(x)$ d'un mot de code est donnée par :

$$c(x) = P(x) + x^{2 \cdot t} \cdot u(x) \tag{2.30}$$

Exemple 2.22.

Suite de l'exemple 2.21.

Soit $\mathbf{u} = \begin{pmatrix} 5 & 3 & 6 \end{pmatrix}$ un mot d'information. Sa représentation polynomiale $u(x)$ est donnée par :

$$u(x) = 5 + 3 \cdot x + 6 \cdot x^2$$

Nous convertissions les symboles de $\mathbf{u}$ en puissances de l'élément primitif α :

$$5 = \begin{pmatrix} 1 & 0 & 1 \end{pmatrix}_2 = \alpha^2 + 1 = \alpha^6,\ 3 = \begin{pmatrix} 0 & 1 & 1 \end{pmatrix}_2 = \alpha + 1 = \alpha^3,\ 6 = \begin{pmatrix} 1 & 1 & 0 \end{pmatrix}_2 = \alpha^2 + \alpha = \alpha^4$$

En utilisant les symboles convertis, $u(x)$ devient :

$$u(x) = \alpha^6 + \alpha^3 \cdot x + \alpha^4 \cdot x^2$$

Le résultat de la multiplication de $u(x)$ par $x^{2\cdot t} = x^4$ sera :

$$x^4 \cdot u(x) = \alpha^6 \cdot x^4 + \alpha^3 \cdot x^5 + \alpha^4 \cdot x^6$$

Lorsqu'on le divise par $g(x) = x^4 + \alpha^3 \cdot x^3 + x^2 + \alpha \cdot x + \alpha^3$, on obtient :

$$\begin{cases} Q(x) = & \alpha^4 \cdot x^2 + \alpha \cdot x + \alpha^6 \\ P(x) = & \alpha^3 \cdot x^3 + \alpha^5 \cdot x + \alpha^2 \end{cases}$$

En utilisant l'équation (2.30), le polynôme $c(x)$ s'écrit :

$$c(x) = \alpha^2 + \alpha^5 \cdot x + \alpha^3 \cdot x^3 + \alpha^6 \cdot x^4 + \alpha^3 \cdot x^5 + \alpha^4 \cdot x^6$$

Ainsi, le résultat du codage de $\mathbf{u}$ par le code RS(7, 3) est le mot de code $\mathbf{c} = \begin{pmatrix}\alpha^2 & \alpha^5 & 0 & \alpha^3 & \alpha^6 & \alpha^3 & \alpha^4\end{pmatrix} = \begin{pmatrix}4 & 7 & 0 & 3 & 5 & 3 & 6\end{pmatrix}$.

Nous présentons maintenant une troisième méthode de construction des mots de code d'un code Reed-Solomon qui a été développée par Reed et Solomon dans leur article original [RS60]. Cette méthode est basée sur l'utilisation de la matrice de Vandermonde comme matrice génératrice. Son principe est détaillé par Wicker et al. dans [SBW94]. Il consiste à multiplier la matrice de Vandermonde par la séquence des mots d'information pour générer les symboles des mots de code.

Un message $\mathbf{u} = \begin{pmatrix}u_1 & u_2 & \cdots & u_k\end{pmatrix}$ avec $u_i \in$GF(q) est usuellement représenté par un polynôme $U(x)$ de degré $k-1$ sous la forme suivante :

$$U(x) = \sum_{i=1}^{k} u_i \cdot x^{i-1} \tag{2.31}$$

Rappelons qu'un corps de Galois de cardinal q peut être défini à l'aide d'un élément primitif, noté α, qui permet de générer tous les éléments non nuls du corps. Ces derniers sont représentés par le vecteur : $\begin{pmatrix}\alpha^0 & \alpha^1 & \cdots & \alpha^{n-1}\end{pmatrix}$ avec $n = q-1$. En notant $\alpha_i = \alpha^i$, un mot de code $\mathbf{c}$ est l'évaluation du polynôme $U(x)$ pour toutes les valeurs α_i :

$$\mathbf{c} = \begin{pmatrix}U(\alpha_0) & U(\alpha_1) & \cdots & U(\alpha_{n-1})\end{pmatrix} \tag{2.32}$$

avec :

$$\begin{aligned} c_1 = U(\alpha_0) &= u_1 + u_2 + u_3 + \cdots + u_k \\ c_2 = U(\alpha_1) &= u_1 + u_2 \cdot \alpha_1 + u_3 \cdot \alpha_1^2 + \cdots + u_k \cdot \alpha_1^{(k-1)} \\ &\vdots \\ c_n = U(\alpha_{n-1}) &= u_1 + u_2 \cdot \alpha_{n-1} + u_3 \cdot \alpha_{n-1}^2 + \cdots + u_k \cdot \alpha_{n-1}^{(k-1)} \end{aligned}$$

Ce système d'équations qui est utilisé pour obtenir les symboles des mots de code $\mathbf{c}$ peut s'écrire sous la forme matricielle définie par l'équation 2.17, tel que la matrice génératrice $\mathbf{G}$ est :

$$\mathbf{G} = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \alpha_1 & \alpha_2 & \cdots & \alpha_{n-1} \\ 1 & \alpha_1^2 & \alpha_2^2 & \cdots & \alpha_{n-1}^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha_1^{k-1} & \alpha_2^{k-1} & \cdots & \alpha_{n-1}^{k-1} \end{pmatrix} \tag{2.33}$$

La forme matricielle de $\mathbf{G}$ est une matrice de Vandermonde. Une matrice de contrôle de parité du

code de Reed-Solomon RS(n, k), $\mathbf{H}$, est définie comme suit [Sha11] :

$$\mathbf{H} = \begin{pmatrix} 1 & \alpha_1 & \alpha_2 & \cdots & \alpha_{n-1} \\ 1 & \alpha_1^2 & \alpha_2^2 & \cdots & \alpha_{n-1}^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha_1^{n-k} & \alpha_2^{n-k} & \cdots & \alpha_{n-1}^{n-k} \end{pmatrix} \tag{2.34}$$

Exemple 2.23.

Reprenons l'exemple du code RS$(7, 3)$ présenté dans l'exemple 2.21. En utilisant l'équation (2.33), la matrice génératrice pour ce code est donnée par :

$$\mathbf{G} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 3 & 6 & 7 & 5 \\ 1 & 4 & 6 & 5 & 2 & 3 & 7 \end{pmatrix}$$

avec $\begin{pmatrix} \alpha_0 & \alpha_1 & \alpha_2 & \alpha_3 & \alpha_4 & \alpha_5 & \alpha_6 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 4 & 3 & 6 & 7 & 5 \end{pmatrix}$.

Nous avons $\mathbf{u} = \begin{pmatrix} 5 & 3 & 6 \end{pmatrix}$, le mot de code $\mathbf{c}$ est obtenu par l'opération suivante :

$$\begin{aligned} \mathbf{c} &= \begin{pmatrix} 5 & 3 & 6 \end{pmatrix} \cdot \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 3 & 6 & 7 & 5 \\ 1 & 4 & 6 & 5 & 2 & 3 & 7 \end{pmatrix} \\ &= \begin{pmatrix} 0 & 6 & 0 & 3 & 3 & 6 & 5 \end{pmatrix} \end{aligned}$$

Le lecteur intéressé pourra trouver en annexe B le tableau de multiplication des éléments du corps GF(2^3) générés par $x^3 + x + 1$.

En utilisant l'équation (2.34), la matrice de contrôle de parité de code est donnée par :

$$\mathbf{H} = \begin{pmatrix} 1 & 2 & 4 & 3 & 6 & 7 & 5 \\ 1 & 4 & 6 & 5 & 2 & 3 & 7 \\ 1 & 3 & 5 & 4 & 7 & 2 & 6 \\ 1 & 6 & 2 & 7 & 4 & 5 & 3 \end{pmatrix}$$

Nous pouvons vérifier que la relation $\mathbf{c} \cdot \mathbf{H}^T = \mathbf{0}$ est satisfaite.

2.3.2 Codes convolutifs

Contrairement aux codes en bloc, la famille des codes convolutifs traitent la séquence d'information par flux sans la découper en blocs finis de bits consécutifs. Cette famille de codes est actuellement la plus employée en raison de leur utilisation dans les turbocodes. Les codes convolutifs ont été inventés en 1954 par Elias [Eli54] sous la forme systématique. Ensuite, Forney [For70] a démontré qu'un bon code convolutif n'est pas nécessairement un code systématique et a proposé une nouvelle construction du code en étudiant sa structure algébrique. Les algorithmes de décodage les plus utilisés actuellement pour les codes convolutifs sont l'algorithme de Viterbi [Vit67] et l'algorithme de BCJR (initiales des inventeurs) [BCJR74] qui est aussi connu par MAP (Maximum A posteriori) ou APP (A Posteriori Probability). Les codes convolutifs sont très présents dans les standards des communications radio mobiles comme les standards de seconde génération, le GSM [3GP05a] et le CDMA-2000 [3GP09], et de troisième génération l'UMTS [3GP05b].

Ryan et Wilson [RW87] sont les premiers à avoir abordé les codes convolutifs non-binaires. Ils ont étudié les codes convolutifs de rendement $1/n$ construits sur les corps GF(q). Ils ont développé dans [RW91] un algorithme de recherche de codes optimaux en termes de capacité de correction

des erreurs. Cet algorithme a été appliqué à deux classes de codes convolutifs non-binaires. La première classe est constituée des codes convolutifs q-aires dans laquelle les séquences d'entrée et de sortie sont composées des symboles qui appartiennent à l'ensemble $\{0, 1, \alpha, \cdots, \alpha^{q-2}\} \in \mathrm{GF}(q)$. En revanche, dans la deuxième classe appelée binaire-à-q-aire, la séquence d'entrée est composée uniquement d'éléments binaires $\{0, 1\}$ du corps GF(2) et la séquence de sortie est composée de symboles du corps GF(q)= $\{0, 1, \alpha, \cdots, \alpha^{q-2}\}$.

La grande différence entre un code en bloc et un code convolutif est l'effet mémoire qui caractérise les codes convolutifs. En effet, un codeur convolutif est conçu autour d'un registre à décalage à μ mémoires. A chaque instant t, le codeur est alimenté par k symboles d'information, $\mathbf{u}(t) = \begin{pmatrix} u_1(t) & u_2(t) & \cdots & u_k(t) \end{pmatrix}$. Il délivre en sortie des séquences infinies de symboles codés de taille n, $\mathbf{c}(t) = \begin{pmatrix} c_1(t) & c_2(t) & \cdots & c_n(t) \end{pmatrix}$, qui dépendent des μ blocs d'information stockés dans le registre à décalage et des symboles présents à l'entrée du codeur. La quantité $K = \mu + 1$ est appelée longueur de contrainte. Un code convolutif, noté $\mathcal{C}(n, k, K)$, est défini par k, la taille du bloc des symboles à l'entrée du codeur, n, la taille du bloc des symboles en sortie du codeur, et K, la longueur de contrainte. Le principe général d'un codeur convolutif à symboles dans GF(q) de rendement k/n et de longueur de contrainte K est décrit sur la figure 2.7.

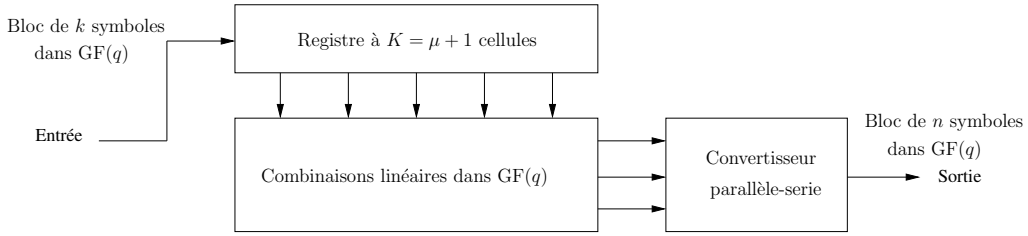


Figure 2.7 — Principe général d'un codeur convolutif $\mathcal{C}(n, k, K)$ dans GF(q)

Deux propriétés peuvent caractériser les codes convolutifs :

- la récursivité signifie que certains symboles de la séquence de sortie du codeur sont réinjectés en entrée.
- systématique signifie que la séquence codée à la sortie du codeur contient le message à transmettre, l'information redondante y est ajoutée.

Deux différentes familles de codes convolutifs sont les plus répandus dans les standards radiomobiles :

- Codes convolutifs de types non-récursifs et non-systématiques, noté NRNSC,
- Codes convolutifs de types récursifs et systématiques, noté RSC.

A partir d'un code de type RSC, il est possible de trouver un code équivalent de type NRNSC [Mar09]. Dans le cadre de notre étude, nous nous sommes intéressés à la famille des codes convolutifs de type NRNSC.

Maintenant, nous illustrons en détail le principe de codage des codeurs convolutifs binaires, puis celui des codes convolutifs généralisés dans GF(q).

2.3.2.1 Codes convolutifs binaires

Les codes convolutifs binaires utilisent uniquement des symboles dans le corps GF(2)= $\{0, 1\}$. Une matrice génératrice peut être représentée sous forme polynomiale ou forme matricielle. Dans ce

mémoire, nous introduirons uniquement la représentation matricielle qui sera utilisée pour réaliser la construction des codes.

Nous noterons $\mathbf{u}$ la séquence d'entrée dans le codeur, tel que :

$$\mathbf{u} = \begin{pmatrix} \mathbf{u}(0) & \mathbf{u}(1) & \cdots \end{pmatrix} \tag{2.35}$$

La matrice génératrice $\mathbf{G}$ de taille $(k \times n)$ pour un code convolutif est sous la forme :

$$\mathbf{G} = \begin{pmatrix} \mathbf{g}_{1,1} & \cdots & \mathbf{g}_{1,n} \\ \vdots & \ldots & \vdots \\ \mathbf{g}_{k,1} & \cdots & \mathbf{g}_{k,n} \end{pmatrix} \tag{2.36}$$

où les coefficients $\mathbf{g}_{i,j}$, $\forall i \in [\![1,k]\!]$ et $\forall j \in [\![1,n]\!]$, sont des vecteurs de longueur K qui contiennent les coefficients des polynômes générateurs. Ils sont tels que :

$$\mathbf{g}_{i,j} = \begin{pmatrix} g_{i,j}(0) & g_{i,j}(1) & \cdots & g_{i,j}(K-1) \end{pmatrix} \tag{2.37}$$

où les coefficients $g_{i,j}(l)$, $\forall l \in [\![0,K-1]\!]$, du polynôme générateur sont 0 ou 1. Les codes convolutifs sont bien adaptés pour la transmission des séquences d'information de longueur quelconque. Soit $\mathbf{c}$ une séquence transmise de mots de code formée :

$$\mathbf{c} = \begin{pmatrix} \mathbf{c}(0) & \mathbf{c}(1) & \cdots \end{pmatrix} \tag{2.38}$$

Chaque bit $c_j(t)$ à l'instant t peut s'exprimer comme une combinaison linéaire des k bits présents à l'entrée du codeur et des $(K-1) \cdot k$ bits contenus dans le registre à décalage :

$$c_j(t) = \sum_{l=0}^{K-1} \sum_{i=1}^{k} u_i(t-l) \cdot g_{i,j}(l), \quad \forall j \in [\![1,n]\!] \tag{2.39}$$

où les opérations d'addition sont faites modulo 2. Alors, le mot de code $\mathbf{c}(t)$ est donné par :

$$\mathbf{c}(t) = \sum_{l=0}^{K-1} \mathbf{u}(t-l) \cdot \mathbf{F}_l \tag{2.40}$$

où les matrices $\mathbf{F}_l$, $\forall l \in [\![0,K-1]\!]$, sont définies par :

$$\mathbf{F}_l = \begin{pmatrix} g_{1,1}(l) & \cdots & g_{1,n}(l) \\ \vdots & \ldots & \vdots \\ g_{k,1}(l) & \cdots & g_{k,n}(l) \end{pmatrix} \tag{2.41}$$

Pour obtenir la séquence de sortie $\mathbf{c}$, le produit de convolution dans l'équation (2.40) peut être remplacé par l'opération de multiplication lorsqu'on utilise une matrice, notée $\mathbf{F}$:

$$\mathbf{F} = \begin{pmatrix} \mathbf{F}_0 & \cdots & \mathbf{F}_{K-1} & & \\ & \ddots & \vdots & \ddots & \\ & & \mathbf{F}_0 & \cdots & \mathbf{F}_{K-1} \\ & & & \ddots & \vdots \\ & & & & \mathbf{F}_0 \end{pmatrix} \tag{2.42}$$

Cette matrice, appelée matrice de codage, est de taille $(k \cdot L \times n \cdot L)$, avec L le nombre de mots

d'information. Ainsi, la séquence de sortie $\mathbf{c}$ est déterminée par :

$$\mathbf{c} = \mathbf{u} \cdot \mathbf{F} \tag{2.43}$$

Exemple 2.24.

Prenons l'exemple d'un codeur de rendement $1/3$ ($k = 1$ et $n = 3$) et de longueur de contrainte $K = 3$, où la matrice génératrice $\mathbf{G}$ est composée de 3 polynômes générateurs $\mathbf{g}_{1,j}$, $\forall j \in \{1, 2, 3\}$:

$$\mathbf{G} = \begin{pmatrix} \mathbf{g}_{1,1} & \mathbf{g}_{1,2} & \mathbf{g}_{1,3} \end{pmatrix} = \begin{pmatrix} 1\,1\,1 & 1\,0\,0 & 0\,0\,1 \end{pmatrix}$$

La matrice de codage $\mathbf{F}$ est composée des K sous-matrices, définies dans l'équation (2.41) :

$$\mathbf{F}_0 = \begin{pmatrix} 1 & 1 & 0 \end{pmatrix} \qquad \mathbf{F}_1 = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} \qquad \mathbf{F}_2 = \begin{pmatrix} 1 & 0 & 1 \end{pmatrix}$$

Soit une séquence d'entrée :

$$\mathbf{u} = \begin{pmatrix} u_1(0) & u_1(1) & u_1(2) & \cdots \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & \cdots \end{pmatrix}$$

Ainsi, la matrice $\mathbf{F}$, décrite dans l'équation (2.42), sera :

$$\mathbf{F} = \begin{pmatrix} \mathbf{F}_0 & \mathbf{F}_1 & \mathbf{F}_2 \\ & \mathbf{F}_0 & \mathbf{F}_1 \\ & & \mathbf{F}_0 \end{pmatrix}$$

En utilisant l'équation (2.43), la séquence de sortie :

$$\mathbf{c} = \begin{pmatrix} c_1(0)\, c_2(0)\, c_3(0) & c_1(1)\, c_2(1)\, c_3(1) & c_1(2)\, c_2(2)\, c_3(2) & \cdots \end{pmatrix}$$

sera obtenue par :

$$\begin{aligned} \mathbf{c} &= \begin{pmatrix} 1 & 1 & 0 & \cdots \end{pmatrix} \cdot \begin{pmatrix} 1\,1\,0 & 1\,0\,0 & 1\,0\,1 \\ & 1\,1\,0 & 1\,0\,0 \\ & & 1\,1\,0 \end{pmatrix} \\ &= \begin{pmatrix} 1\,1\,0 & 0\,1\,0 & 0\,0\,1 & \cdots \end{pmatrix} \end{aligned}$$

2.3.2.2 Codes convolutifs non-binaires

Dans cette partie, une généralisation des codes convolutifs dans GF(2) aux codes dans GF(q) est présentée. Le schéma d'implémentation d'un code convolutif de rendement k/n dans GF(q) avec une longueur de contrainte K est représenté sur la figure 2.8.

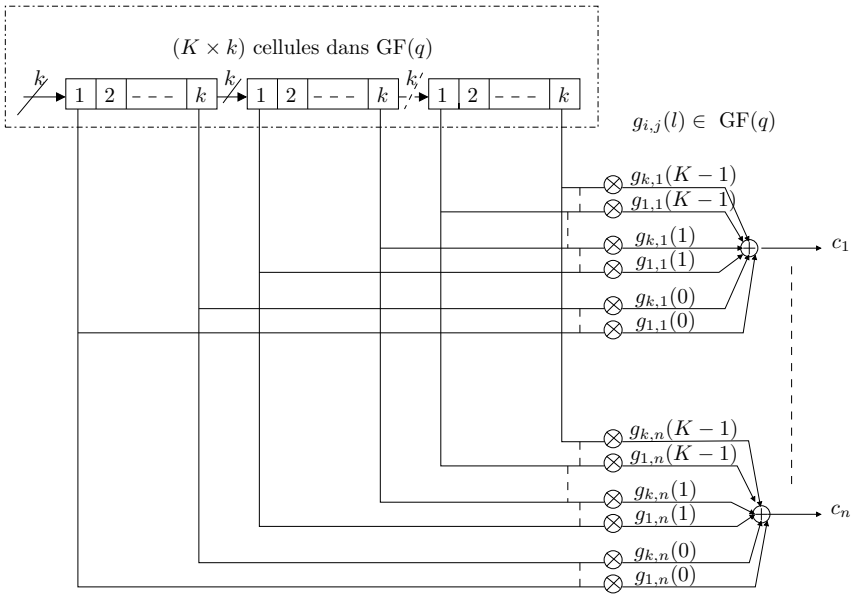


Figure 2.8 — Schéma d'implémentation d'un code convolutif non-binaire de rendement k/n

Ce codeur est composé d'un ensemble de registres à décalage à $k \cdot K$ cellules dans GF(q), de n opérations d'addition dans GF(q) et de $n \cdot k \cdot K$ coefficients $g_{i,j}(l) \in$ GF(q). Les coefficients $g_{i,j}(l)$ correspondent aux éléments de la matrice génératrice $\mathbf{G}$. Après avoir multiplié les contenus des cellules par les coefficients $g_{i,j}(l)$, les opérations d'addition peuvent être appliquées dans GF(q). La matrice de codage $\mathbf{F}$ des codes convolutifs non-binaires a la même forme que celle des codes convolutifs binaires décrite dans l'équation (2.42). Chaque élément $\mathbf{F}_l, \forall l \in [\![0, \cdots, K-1]\!]$, est une sous matrice non-binaire de taille $(k \times n)$. Chaque ligne de $\mathbf{F}$ s'obtient en décalant la précédente de n colonnes. Cela explique le principe de fonctionnement d'un code convolutif qui s'appuie sur la notion de registres à décalage. Les k symboles d'information $\mathbf{u}(t) = \begin{pmatrix} u_1(t) & u_2(t) & \cdots & u_k(t) \end{pmatrix}$, avec $u_i(t) \in$ GF(q), sont introduits en parallèle dans le codeur à un instant t. Ces symboles vont être décalés temporellement par bloc de k symboles à l'aide de registres à décalage. La sortie du codeur $\mathbf{c}(t) = \begin{pmatrix} c_1(t) & c_2(t) & \cdots & c_n(t) \end{pmatrix}$, avec $c_j(t) \in$ GF(q), à l'instant t dépend de ses entrées à cet instant et de l'état des registres. L'opération de codage des codes convolutifs dans GF(q) est réalisée en utilisant l'équation (2.43).

Un code convolutif dans GF(q) de rendement k/n possède $k \cdot (K-1)$ registres à décalage qui peuvent prendre $q^{k \cdot (K-1)}$ états différents. Puisque l'entrée est constituée de k symboles dans GF(q), il existe alors q^k transitions possibles à partir d'un état du codeur à un instant donné.

Exemple 2.25.

Prenons l'exemple du $\mathcal{C}(3,1,3)$ code convolutif dans GF(2^2) défini par le polynôme primitif $p(x) = x^2 + x + 1$. La séquence d'entrée $\mathbf{u}$ et la séquence de sortie $\mathbf{c}$ sont des éléments de GF(2^2)= $\{0, 1, \alpha, \alpha^2\} = \{0, 1, 2, 3\}$. La matrice génératrice $\mathbf{G}$ du code est ici :

$$\mathbf{G} = \begin{pmatrix} \mathbf{g}_{1,1} & \mathbf{g}_{1,2} & \mathbf{g}_{1,3} \end{pmatrix} = \begin{pmatrix} 1\,2\,3 & 2\,2\,2 & 3\,2\,1 \end{pmatrix}$$

La matrice de codage $\mathbf{F}$ est composée de K sous-matrices qui s'écrivent :

$$\mathbf{F}_0 = \begin{pmatrix} 1 & 2 & 3 \end{pmatrix} \qquad \mathbf{F}_1 = \begin{pmatrix} 2 & 2 & 2 \end{pmatrix} \qquad \mathbf{F}_2 = \begin{pmatrix} 3 & 2 & 1 \end{pmatrix}$$

La séquence $\mathbf{u}$ à l'entrée du codeur est :

$$\mathbf{u} = \begin{pmatrix} 2 & 3 & 2 \end{pmatrix}$$

En utilisant la matrice de codage $\mathbf{F}$ décrite dans l'équation (2.42), la séquence $\mathbf{c}$ à la sortie du codeur est obtenue par :

$$\begin{aligned}\mathbf{c} &= \begin{pmatrix}2 & 3 & 2\end{pmatrix} \cdot \begin{pmatrix}1\,2\,3 & 2\,2\,2 & 3\,2\,1 \\ & 1\,2\,3 & 2\,2\,2 \\ & & 1\,2\,3\end{pmatrix} \\ &= \begin{pmatrix}2\,3\,1 & 0\,2\,1 & 2\,1\,2\end{pmatrix}\end{aligned}$$

A chaque polynôme générateur $\mathbf{g}_{i,j}$, on peut associer sa transformée en D définie par :

$$\mathbf{g}_{i,j}(D) = \sum_{l=0}^{K-1} g_{i,j}(l) \cdot D^l \tag{2.44}$$

Nous notons μ_i la mémoire de la i-ème entrée du codeur. La mémoire μ_i est déterminée par :

$$\mu_i = \max_{j=1,\cdots,n} (\deg \mathbf{g}_{i,j}(D)) \tag{2.45}$$

Nous rappelons qu'un mot de code à la sortie du codeur dépend du mot d'information présent à l'entrée du codeur et des $K-1=\mu$ mots ayant été introduits précédemment. Alors, la mémoire μ peut être définie par le nombre de mots d'information gardés en mémoire dans les registres à décalage. Elle correspond à :

$$\mu = K - 1 = \max_{i=1,\cdots,k} (\mu_i) \tag{2.46}$$

Nous avons vu précédemment que la matrice de contrôle de parité $\mathbf{H}$ est indispensable lors de l'opération de décodage afin de détecter et/ou corriger les erreurs de transmission. Cette matrice est considérée comme une matrice génératrice du code dual $\mathcal{C}^{\perp}$. Notons $\mu^{\perp}$ la mémoire du code dual d'un code convolutif $\mathcal{C}(n,k,K)$ qui est caractérisé par une mémoire μ. La relation entre les deux mémoires est définie par :

$$\mu = \left\lceil \frac{\mu^{\perp}}{k} \right\rceil \tag{2.47}$$

avec :

$$\mu^{\perp} = \sum_{i=1}^{k} \mu_i \tag{2.48}$$

Une matrice de contrôle de parité $\mathbf{H}$ de taille $((n-k)\cdot L \times n \cdot L)$ sera composée de $\mu^{\perp}+1$ sous matrices $\mathbf{H}_i$, $i \in [\![0, \mu^{\perp}]\!]$, de taille $((n-k) \times n)$, telle que :

$$\mathbf{H} = \begin{pmatrix} \mathbf{H}_0 & & & & & \\ \mathbf{H}_1 & \mathbf{H}_0 & & & & \\ \vdots & \ddots & \ddots & & & \\ \mathbf{H}_{\mu^{\perp}} & \cdots & \mathbf{H}_1 & \mathbf{H}_0 & & \\ & \mathbf{H}_{\mu^{\perp}} & \cdots & \mathbf{H}_1 & \mathbf{H}_0 & \\ & & \ddots & \ddots & \ddots & \ddots \end{pmatrix} \tag{2.49}$$

où :

$$\mathbf{H}_i = \begin{pmatrix} h_{1,1}(i) & \cdots & h_{1,k}(i) & h_0(i) & & \\ \vdots & \cdots & \vdots & & \ddots & \\ h_{n-k,1}(i) & \cdots & h_{n-k,k}(i) & & & h_0(i) \end{pmatrix} \tag{2.50}$$

Une séquence $\mathbf{c}$ de longueur $n \cdot L$ est une séquence de mots de code si elle vérifie :

$$\mathbf{c} \cdot \mathbf{H}^T = 0 \tag{2.51}$$

Par conséquent, la matrice $\mathbf{H}$ et la matrice de codage $\mathbf{F}$ vérifient la relation :

$$\mathbf{F} \cdot \mathbf{H}^T = \mathbf{0} \tag{2.52}$$

Exemple 2.26.

Reprenons l'exemple du $\mathcal{C}(3,1,3)$ dans GF(2^2) de l'exemple 2.25. Les polynômes générateurs de ce code ont le même degré 2. D'après les équations (2.45) et (2.46), la mémoire globale μ du code est égale à 2. En utilisant la relation (2.48), la mémoire du code dual est $\mu^{\perp} = 2$.

Une matrice de contrôle de parité est composée de 3 sous matrices de taille (2×3), tel que :

$$\mathbf{H}_0 = \begin{pmatrix} 2 & 1 & 0 \\ 3 & 0 & 1 \end{pmatrix} \qquad \mathbf{H}_1 = \begin{pmatrix} 1 & 0 & 0 \\ 3 & 0 & 0 \end{pmatrix} \qquad \mathbf{H}_2 = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 0 & 0 \end{pmatrix}$$

En posant $L = 3$, cette matrice sera de taille (6×9) :

$$\mathbf{H} = \begin{pmatrix} 2\,1\,0 & & \\ 3\,0\,1 & & \\ 1\,0\,0 & 2\,1\,0 & \\ 3\,0\,0 & 3\,0\,1 & \\ 1\,0\,0 & 1\,0\,0 & 2\,1\,0 \\ 2\,0\,0 & 3\,0\,0 & 3\,0\,1 \end{pmatrix} \tag{2.53}$$

On peut vérifier que la matrice $\mathbf{H}$ obtenue satisfait la relation (2.52).

2.4 Conclusion

Dans ce chapitre, nous avons tout d'abord introduit le fonctionnement d'une chaîne de transmission numérique, de la source d'information jusqu'au destinataire. Les modèles de canaux de transmission que nous utiliserons dans cette thèse ont été décrits, en particulier les canaux BSC et QSC sur lesquels nos algorithmes d'identification aveugle seront tout d'abord appliqués.

L'objectif de nos travaux étant d'améliorer les performances des algorithmes d'identification aveugle des paramètres des codes correcteurs d'erreurs binaires déjà existants et de les généraliser aux codes correcteurs d'erreurs non-binaires, la seconde partie de ce chapitre a été consacrée à l'étude des propriétés des codes binaires et non-binaires que nous abordons dans cette thèse. Pour chaque code, l'opération de codage a été décrite dans le souci de présenter des paramètres indispensables pour réaliser l'opération inverse, le décodage. L'identification aveugle de ces paramètres fait l'objet du troisième chapitre en supposant que les données reçues ne sont pas entachées d'erreurs.

CHAPITRE 3

Identification aveugle des codes dans le cas non-bruité

3.1 Introduction

Dans un contexte de transmission non-coopératif, comme l'interception militaire ou la radio cognitive, le récepteur ne connait pas les paramètres des codes correcteurs d'erreurs utilisés à l'émission. Il devra les estimer d'une manière autodidacte à travers la seule connaissance des données reçues. A ce jour, certains travaux ont été menés sur cette thématique. Ces travaux se sont limités à des algorithmes et des codes travaillant dans le corps de Galois binaire GF(2). Dans cette thèse, notre objectif vise à développer des algorithmes d'identification plus généraux adaptés aux codes correcteurs d'erreurs non-binaires. Dans ce chapitre, une transmission parfaite est considérée afin de tester la faisabilité de notre algorithme. De ce fait, nous avons adapté l'algorithme de reconnaissance des codes convolutifs binaires développé par Marazin dans sa thèse [Mar09] au contexte des codes non-binaires. Cet algorithme est basé sur l'utilisation du critère du rang dont les comportements permettent l'identification des paramètres du code. Cependant, ce critère a été exploité sans être théoriquement justifié. Ainsi, nous proposons une étude théorique et algébrique plus poussée afin de justifier l'utilisation des comportements théoriques du rang, ainsi que des cas particuliers pouvant apparaître pour des paramètres spécifiques de codes.

Ce chapitre est organisé comme suit : d'abord on introduira la technique d'identification aveugle des codes binaires basée sur le critère du rang dans la section 3.2, ensuite nous présenterons l'étude théorique sur les comportements du rang en section 3.3. Dans la section 3.4, l'algorithme d'identification des codes non-binaires sera présenté. L'identification aveugle d'une base du code dual des codes dans GF(2^m) sera développée en section 3.5. Enfin, la section 3.6 conclut le chapitre.

3.2 Identification des codes binaires

Les travaux sur la thématique d'identification aveugle des paramètres du bloc de codage canal se sont limités à GF(2). Dans le cadre d'une transmission non-bruitée, il existe quelques algorithmes permettant d'identifier en aveugle les paramètres d'un code convolutif. Cela a fait l'objet du travail de Rice dans [Ric95] lorsqu'il a développé une approche pour identifier les codes convolutifs de rendement $1/2$. Cette approche a été ensuite généralisée par Filiol au cas des codes de rendement $1/n$ [Fil97], puis aux codes de rendement $n-1/n$ [Fil01] et aussi au cas des codes convolutifs poinçonnés [Fil00]. Barbier a proposé dans [Bar05] un algorithme de reconstruction aveugle des codes convolutifs de rendement k/n dans le cas d'un schéma de turbocode particulier où les deux codeurs sont identiques. Dans [MGB09a], une méthode permettant d'identifier les paramètres d'un code convolutif ainsi qu'une nouvelle approche capable d'identifier le second codeur d'un turbo-

code ont été développées. En parallèle, l'identification aveugle des codes en bloc linéaires a été étudiée. Dans [Pla96], une technique basée sur des algorithmes de recherche des mots de code à faible poids de parité [Can98a, Ste89a] a été proposée. Cette technique a été améliorée par Valembois dans [Val01] en utilisant des testes statistiques et récemment par Cluzeau dans [Clu06b]. Une deuxième technique basée sur l'utilisation du critère du rang a été développée par Burel et al. dans [BG03] pour un canal de transmission non-bruitée. Cette technique a été utilisée pour identifier en aveugle les paramètres des entrelaceurs et des codes en bloc binaires, puis a été adaptée dans [Mar09] aux codes convolutifs binaires. Le principe de cette technique sera présenté dans la section 3.2.1.

3.2.1 Principe de la méthode basée sur le critère du rang

Nous présentons dans cette section le principe de l'algorithme d'identification aveugle des paramètres des codes binaires. La plupart des travaux traitant le cas des codes binaires sont basés sur le calcul du rang de matrices construites à partir des données reçues. Dans [MGB09a], le critère du rang a été appliqué pour détecter la présence du code et identifier les paramètres des codes convolutifs dans GF(2). Pour illustrer le principe de la méthode basée sur ce critère, les codes convolutifs $\mathcal{C}(n, k, K)$ dans GF(2) sont considérés.

En supposant que les données reçues sont synchronisées et non entachées d'erreurs, ces données sont réorganisées sous forme de matrices, notées $\mathbf{R}_l$, de taille $(M \times l)$. Le nombre de colonnes l varie entre 1 et l_{max} et le nombre de lignes M dépendant de l est déterminé par $M = \lfloor L/l \rfloor$, avec L la taille des données reçues. Un exemple de réorganisation des données reçues sous forme de matrice $\mathbf{R}_l$, avec $l = 3$, est représenté sur la figure 3.1.

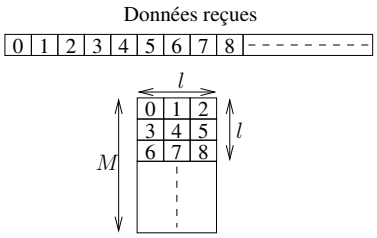


Figure 3.1 — Exemple de matrice $\mathbf{R}_l$ avec $l = 3$

Pour chaque matrice $\mathbf{R}_l$ construite, le rang est calculé afin de déterminer l'ordre de la dépendance. Le rang est défini par le nombre de lignes ou de colonnes qui sont linéairement indépendantes. Ainsi, si toutes les matrices $\mathbf{R}_l$ présentent des rangs pleins, nous pouvons en conclure qu'aucun code n'a été utilisé. Par contre, l'existence de chutes de rang dans quelques matrices indique la présence d'un code. Par ailleurs, le comportement du rang permet d'identifier des paramètres du code correcteur d'erreurs utilisé.

3.2.2 Identification des paramètres

Le rang des matrices $\mathbf{R}_l$ présente généralement deux comportements différents en fonction de l, pour $\alpha \in \mathbb{N}$:

- Si $l = \alpha \cdot n$ et $l \geq n_a$, $\alpha \in \mathbb{N}$

$$\text{rang}(\mathbf{R}_l) = l \cdot \frac{k}{n} + \mu^{\perp} < l \tag{3.1}$$

- Si $l \neq \alpha \cdot n$ ou $l < n_a$, $\alpha \in \mathbb{N}$

$$\text{rang}(\mathbf{R}_l) = l \tag{3.2}$$

où n_a représente la taille de la première matrice de rang déficient et $\mu^\perp$ est la mémoire du code dual d'un code convolutif. La mémoire $\mu^\perp$ est nulle pour les codes en blocs. Dans le cas des codes convolutifs, elle est déterminée par :

$$\mu^\perp = \sum_{i=1}^{k} \mu_i \tag{3.3}$$

La mémoire globale du code est reliée à la mémoire du code dual et à la longueur de contrainte du code par :

$$\mu = \left\lfloor \frac{\mu^\perp}{k} \right\rfloor = K - 1 \tag{3.4}$$

L'équation (3.1) montre que les valeurs des chutes de rang dépendent des paramètres du code. De ce fait, connaissant seulement le rang de deux matrices consécutives de rang déficient, on pourra identifier tous les paramètres du code qui sont listés ci-dessous.

- Identification de la taille des mots de code n

La différence entre deux valeurs de l correspondant à deux matrices consécutives de rang déficient définie la taille des mots de code.

- Identification de la taille des mots d'information k

La différence entre deux valeurs consécutives de rang déficient de $\mathbf{R}_l$ représente la taille des mots d'information.

- Identification de la longueur de contrainte K

En ayant identifié les paramètres n et k au préalable, à partir de l'équation (3.1), il est possible d'estimer $\mu^\perp$ en utilisant une seule matrice de rang déficient. Puis, la mémoire du code μ et la longueur de contrainte K pourront être déterminées par l'équation (3.4).

Exemple 3.27.

Pour illustrer le principe de la méthode d'identification, nous prenons l'exemple du $\mathcal{C}(4,2,2)$ code convolutif dont la matrice de codage est définie par deux sous-matrices :

$$\mathbf{F}_0 = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}, \mathbf{F}_1 = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix}$$

Le rang des matrices $\mathbf{R}_l$ pour $l = 1, \cdots, 25$ est représenté sur la figure 3.2. Pour ce code, la taille de la première matrice de rang déficient est $n_a = \alpha \cdot n = 8$. On peut vérifier que les matrices de taille $l = \alpha \cdot n > 8$ présentent des déficiences de rang. D'après l'équation (3.1), les paramètres du code peuvent être identifiés en utilisant deux matrices consécutives de rang déficient. Par exemple, en utilisant $\mathbf{R}_{12}$ et $\mathbf{R}_{16}$, nous obtenons :

$$\begin{cases} n = 16 - 12 = 4 \\ k = \text{rang}(\mathbf{R}_{16}) - \text{rang}(\mathbf{R}_{12}) = 10 - 8 = 2 \\ \mu^\perp = \text{rang}(\mathbf{R}_{12}) - 12 \cdot \dfrac{k}{n} = 8 - 12 \cdot \dfrac{2}{4} = 2 \\ \mu = \left\lfloor \dfrac{\mu^\perp}{k} \right\rfloor = \left\lfloor \dfrac{2}{2} \right\rfloor = 1 \\ K = \mu + 1 = 1 + 1 = 2 \end{cases}$$

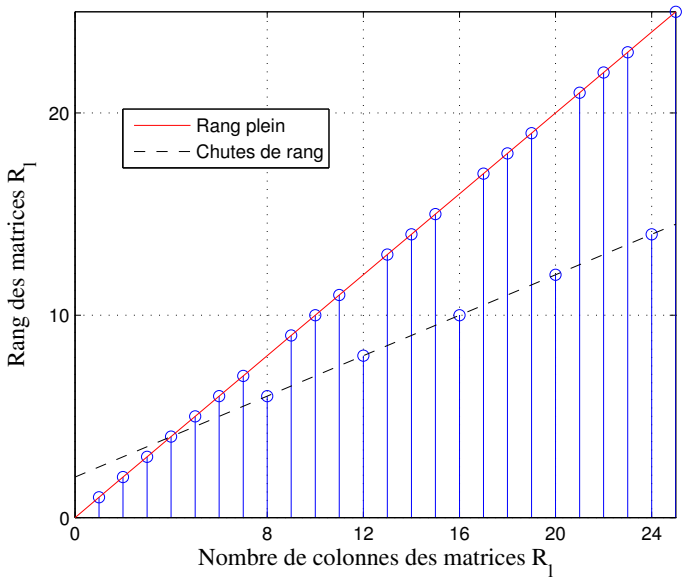


Figure 3.2 — Rang des matrices $\mathbf{R}_l$ pour le code convolutif $\mathcal{C}(4, 2, 2)$

3.3 Étude du comportement du critère du rang

La plupart des techniques proposées dans la littérature pour identifier en aveugle les paramètres des codes correcteurs d'erreurs sont basées sur le calcul du rang des matrices construites à partir des données reçues. Les chutes de rang observées pour certaines matrices ont été exploitées, sans démonstration théorique, pour déterminer les paramètres du code. Dans ce contexte, nous proposons dans cette section une étude théorique et algébrique du comportement du critère du rang, ainsi que des cas particuliers pouvant apparaître pour des paramètres spécifiques de codes. Notons que l'étude proposée dans cette section a fait l'objet d'une publication dans la revue MTA Review [ZGM+12a] et dans la conférence COMM 2012 [ZGM+12b].

3.3.1 Propriétés du comportement dominant du rang

On peut remarquer à travers l'exemple 3.27 que la première matrice de taille n_a est déterminée jusqu'ici empiriquement. Dans [MGB11], la valeur de n_a a été déterminée en supposant que les équations (3.1) et (3.2) aient été démontrées. L'objectif de cette partie est de justifier théoriquement ces équations ainsi que la valeur n_a.

Le rang d'une matrice est le nombre de lignes ou de colonnes qui sont linéairement indépendantes. Dans le cadre de cette étude, il est défini par :

$$\text{rang}(\mathbf{R}_l) = \min(l, \text{nombre maximum de colonnes linéairement indépendantes}) \tag{3.5}$$

Prenons le cas général d'un code convolutif (voir la figure 2.8) qui est composé d'un ensemble de registres à décalage. A un instant donné, les k bits d'information seront introduits en parallèle dans le codeur. Ces bits seront ensuite décalés par bloc de k bits à l'aide de registres à décalage. A l'instant t, la sortie du codeur dépend des k bits d'entrées et de l'état des $(k-1)$ ou μ registres. Ainsi, un mot de code de taille n dépend de $(\sum_{i=1}^{k} \mu_i + k = \mu^{\perp} + k)$ bits d'information qui sont composés des k bits d'entrée à l'instant t et des $\mu^{\perp}$ bits ayant été introduits précédemment. Pour générer α mots de code, il faut $\alpha \cdot (\mu^{\perp} + k)$ bits d'information. De ce fait, α mots de code consécutifs

dépendent de $(\mu^{\perp} + \alpha \cdot k)$ bits d'information distincts.

Par conséquent, pour $l = \alpha \cdot n$, le rang des matrices $\mathbf{R}_l$ construites avec les mots de code est donné par :

$$\text{rang}(\mathbf{R}_{\alpha \cdot n}) = \min(\alpha \cdot n, \mu^{\perp} + \alpha \cdot k) \tag{3.6}$$

Nous traçons sur la figure 3.3 les droites décrivant les équations $\alpha \cdot n$ et $\mu^{\perp} + \alpha \cdot k$. Nous noterons a le point d'intersection des deux droites $\alpha \cdot n$ et $\mu^{\perp} + \alpha \cdot k$, pour $\alpha \in \mathbb{N}$. Ce point est déterminé par :

$$a = \frac{\mu^{\perp}}{n - k} \tag{3.7}$$

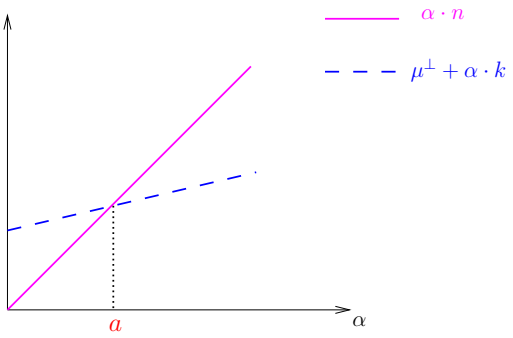


Figure 3.3 — Intersection des droites décrivant les comportements du rang

Le principe des codes correcteurs d'erreurs est d'introduire de la redondance aux bits d'information, ce qui implique que $n > k$ et $k > 0$. De ce fait, pour tout $\alpha > a$, nous aurons $\alpha \cdot n > \mu^{\perp} + \alpha \cdot k$ puisque ces deux droites sont croissantes et la pente de la droite $\alpha \cdot n$ est supérieure à celle de la droite $\mu^{\perp} + \alpha \cdot k$. Par conséquent, trouver la première chute de rang visible ou la chute de rang minimale revient à déterminer la valeur entière minimale de α, notée α_{min}, qui est strictement supérieure à a car toutes les matrices $\mathbf{R}_{\alpha \cdot n}$, pour $\alpha \leq a$, sont de rang plein.

En fait, $\alpha_{min} = \lfloor a + 1 \rfloor = \lfloor a \rfloor + 1 = \lfloor \mu^{\perp}/(n-k) + 1 \rfloor$. Cette valeur est bien strictement supérieur à a. En effet, si $\mu^{\perp}$ est divisible par $(n - k)$, alors a est un entier et α_{min} sera égale à $a + 1$ qui est strictement supérieur à a. Par conséquent, on peut détecter la première chute de rang minimale en considérant :

$$\alpha_{min} = \lfloor a + 1 \rfloor \tag{3.8}$$

Le lecteur intéressé pourra se référer à l'annexe C.1 où une démonstration détaillée de cette formule est exposée. Donc, la taille de la première matrice de rang déficient, notée n_a, est déterminée par :

$$n_a = \alpha_{min} \cdot n = \left\lfloor \frac{\mu^{\perp}}{n - k} + 1 \right\rfloor \cdot n \tag{3.9}$$

Exemple 3.28.

Nous reprenons l'exemple 3.27 pour illustrer que l'utilisation de l'équation (3.9) nous permet d'obtenir la même taille n_a :

$$n_a = \left\lfloor \frac{2}{2} + 1 \right\rfloor \cdot 4 = 8 \tag{3.10}$$

Dans la section suivante, nous montrerons que le comportement du rang est plus complexe que le comportement dominant décrit dans la sous-section 3.2.1. En fait, le comportement du rang dépend aussi de la matrice génératrice utilisée pour le codage et plus finement des paramètres du

code. Ainsi, des chutes de rang supplémentaires pour quelques tailles de matrices spécifiques non multiples de la taille des mots de code sont également présentées.

3.3.2 Impact de la matrice génératrice du code sur la formule du rang

Dans cette partie, notre objectif est de montrer que la formule du rang donnée par l'équation (3.1) n'est pas respectée par certains codes ayant des paramètres spécifiques. En effet, en fonction de la matrice génératrice utilisée pour le codage, les matrices $\mathbf{R}_l$ peuvent avoir certaines chutes de rang pour $\alpha \cdot n < n_a$. Pour expliquer ce phénomène, un exemple d'un code convolutif avec des paramètres spécifiques est présenté.

Exemple 3.29.

Prenons l'exemple du $\mathcal{C}(6,3,3)$ code convolutif défini par la matrice de codage $\mathbf{F}$ composée de trois sous-matrices :

$$\mathbf{F}_0 = \begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}, \mathbf{F}_1 = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}, \mathbf{F}_2 = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

D'après la formule du rang de l'équation (3.1), la première matrice de rang déficient est de taille :

$$n_a = \alpha_{min} \cdot n = \left\lfloor \frac{\mu^{\perp}}{n-k} + 1 \right\rfloor \cdot n = \left\lfloor \frac{6}{6-3} + 1 \right\rfloor \cdot 6 = 18 \tag{3.11}$$

et le rang de cette matrice est :

$$\text{rang}(\mathbf{R}_{n_a}) = \mu^{\perp} + \alpha_{min} \cdot k = 6 + 3 \cdot 3 = 15 \tag{3.12}$$

Le rang des matrices $\mathbf{R}_l$ pour $l = 1, \cdots, 45$ est représenté sur la figure 3.4. On peut remarquer que la matrice $\mathbf{R}_6$ et $\mathbf{R}_{12}$ présentent une déficience de rang :

$$\begin{cases} \text{rang}(\mathbf{R}_6) = 5 \\ \text{rang}(\mathbf{R}_{12}) = 10 \end{cases}$$

Or, d'après l'équation (3.1), ces matrices devraient être de rang plein si le comportement dominant du rang avait été respecté. Cette chute de rang est uniquement liée à l'existence de combinaisons linéaires particulières dans la matrice de codage. En effet, la matrice de codage équivalente qui nous permet de construire la matrice $\mathbf{R}_6$ est donnée par :

$$\mathbf{F}^{(6)} = \begin{pmatrix} \mathbf{F}_2 \\ \mathbf{F}_1 \\ \mathbf{F}_0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

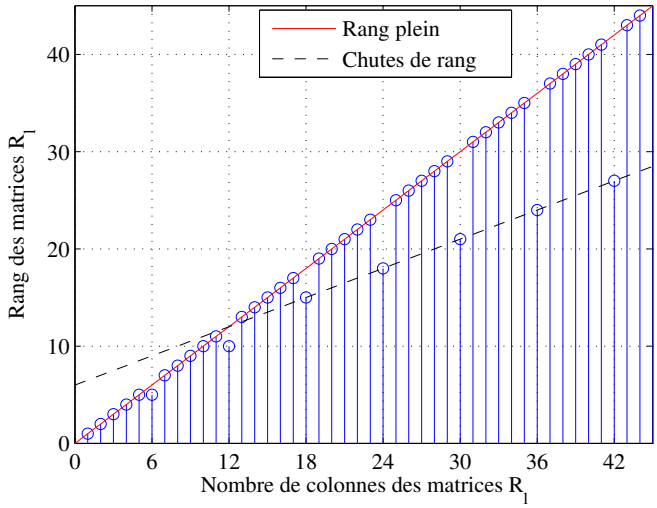


Figure 3.4 — Rang des matrices $\mathbf{R}_l$ pour le code convolutif $\mathcal{C}(6,3,3)$

On peut vérifier que la matrice $\mathbf{F}^{(6)}$ n'est pas de rang plein car elle contient la combinaison linéaire suivante :

$$\mathbf{f}_1^{(6)} + \mathbf{f}_2^{(6)} = \mathbf{f}_3^{(6)}$$

où $\mathbf{f}_i^{(6)}$, $\forall i = 1, \cdots, 6$, représente la i-ème colonne de la matrice $\mathbf{F}^{(6)}$.

Pour construire la matrice $\mathbf{R}_{12}$, la matrice de codage équivalente, notée $\mathbf{F}^{(12)}$, est donnée par :

$$\mathbf{F}^{(12)} = \begin{pmatrix} \mathbf{F}_2 & \mathbf{0} \\ \mathbf{F}_1 & \mathbf{F}_2 \\ \mathbf{F}_0 & \mathbf{F}_1 \\ \mathbf{0} & \mathbf{F}_0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix} \tag{3.13}$$

avec $\mathbf{0}$ une matrice nulle de taille $(k \times n)$. On peut vérifier qu'il existe deux combinaisons linéaires entre les colonnes de la matrice $\mathbf{F}^{(12)}$:

$$\begin{cases} \mathbf{f}_1^{(12)} + \mathbf{f}_2^{(12)} = \mathbf{f}_3^{(12)} \\ \mathbf{f}_7^{(12)} + \mathbf{f}_8^{(12)} = \mathbf{f}_9^{(12)} \end{cases}$$

où $\mathbf{f}_i^{(12)}$, $\forall i = 1, \cdots, 12$, représente la i-ème colonne de la matrice $\mathbf{F}^{(12)}$.

A travers l'exemple ci-dessus, on peut déduire que les petites chutes de rang supplémentaires pour certaines matrices $\mathbf{R}_{\alpha \cdot n}$ avec $\alpha \cdot n < n_a$ peuvent être liées à la matrice de codage.

3.3.3 Étude des chutes de rang multiples

Comme mentionné dans la section 3.2, les chutes de rang pour les matrices $\mathbf{R}_l$ nous permettent d'identifier les paramètres des codes convolutifs et des codes en bloc (où $\mu^{\perp} = 0$). Ces chutes de rang dominantes, définies par l'équation (3.1), sont représentées par une seule droite de pente k/n tel que son intersection avec l'axe des ordonnés est la mémoire du code dual $\mu^{\perp}$. En revanche, il existe des codes avec des paramètres spécifiques qui génèrent plusieurs droites de pentes différentes. En fait, le nombre de ces droites dépend des paramètres du code. Mais, la droite permettant d'identifier les bons paramètres du code est toujours présente. Nous examinons en détails les matrices $\mathbf{R}_l$ des codes convolutifs qui permettent de générer plusieurs droites de chutes de rang.

Une matrice $\mathbf{R}_l$ de taille $(M \times l)$ est construite à partir des mots de code de taille n. Le nombre de colonnes peut être exprimé en fonction de n par :

$$l = q \cdot n + r,\ r < n \tag{3.14}$$

où le quotient q défini par $q = \lfloor l/n \rfloor$ correspond au nombre de mots de code et le reste de la division $r = \mathrm{mod}(l, n)$ correspond au nombre de bits supplémentaires. Notre idée consiste à décomposer la matrice $\mathbf{R}_l$ en β sous-matrices, notées $\mathbf{R}_{l,i}$, $\forall i \in [\![0, \beta - 1]\!]$, de taille $(\lfloor M/\beta \rfloor \times l)$ avec $\beta = n/d$ où d est le plus grand commun diviseur (PGCD) de l et n :

$$d = \mathrm{pgcd}(l, n) \tag{3.15}$$

La figure 3.5 représente un exemple de décomposition de la matrice $\mathbf{R}_l$. Toutes les lignes de la matrice $\mathbf{R}_{l,i}$ ont le même nombre de bits contenus dans le premier bloc (représenté sur la figure 3.5 par la couleur gris foncé), noté $a_{l,i}$, $\forall i \in [\![0, \beta - 1]\!]$, dont la taille est inférieure à n. Elles ont également le même nombre de bits contenus dans le dernier bloc (représenté sur la figure 3.5 par la couleur gris clair), noté $b_{l,i}$, $\forall i \in [\![0, \beta - 1]\!]$, dont la taille est inférieure à n.

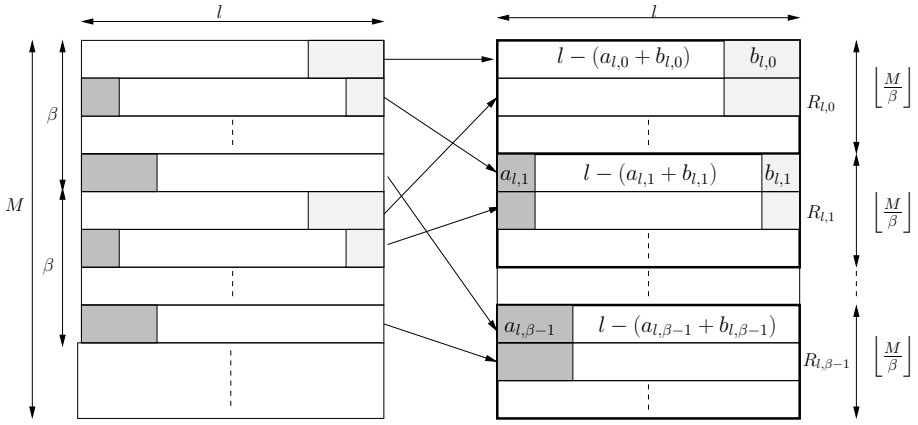


Figure 3.5 — Exemple de décomposition de la matrice $\mathbf{R}_l$

Les valeurs de $a_{l,i}$ et $b_{l,i}$ sont définies par :

$$a_{l,i} = n - b_{l,i-1},\ \forall i \in [\![1, \beta - 1]\!] \tag{3.16}$$

$$b_{l,i} = \mathrm{mod}(r - a_{l,i}, n),\ \forall i \in [\![0, \beta - 1]\!] \tag{3.17}$$

avec $a_{l,0} = 0$ et $b_{l,0} = r$. Le rang des matrices $\mathbf{R}_l$ peut s'écrire :

$$\mathrm{rang}(\mathbf{R}_l) = \min(l, \varphi(l)) \tag{3.18}$$

avec :

$$\varphi(l) = \sum_{i=0}^{\beta-1} \left[\min(a_{l,i}, k) + \min(b_{l,i}, k) + \frac{l - (a_{l,i} + b_{l,i})}{n} \cdot k + \mu^{\perp} \right] \tag{3.19}$$

Notre objectif est maintenant de démontrer et de simplifier l'expression de $\varphi(l)$ donnée par l'équation (3.19). Le rang global des matrices $\mathbf{R}_l$ peut être exprimé en fonction de la somme des rangs des sous-matrices $\mathbf{R}_{l,i}$ par :

$$\text{rang}(\mathbf{R}_l) = \min \left(l, \sum_{i=0}^{\beta-1} \text{rang}(\mathbf{R}_{l,i}) \right) \tag{3.20}$$

où le rang des sous-matrices $\mathbf{R}_{l,i}$ est défini par :

$$\text{rang}(\mathbf{R}_{l,i}) = \text{beg}_{l,i} + s_{l,i} + \text{end}_{l,i} \tag{3.21}$$

avec :

- $\text{beg}_{l,i}$: le nombre de bits d'information récupérés par le bloc $a_{l,i}$
- $s_{l,i}$: le nombre de bits d'information récupérés par le bloc $(l - (a_{l,i} + b_{l,i}))$ des mots de code complets
- $\text{end}_{l,i}$: le nombre de bits d'information récupérés par le bloc $b_{l,i}$

Nombre de bits d'information $s_{l,i}$

Rappelons tout d'abord qu'un mot de code dépend des k bits d'entrée à l'instant t et des $\mu^{\perp}$ bits ayant été introduits précédemment. En faisant l'hypothèse que nous avons suffisamment de mots de code pour récupérer les $\mu^{\perp}$ bits d'information précédents (pour $l \geq n_a$), $s_{l,i}$ est déterminé par :

$$s_{l,i} = k \cdot (\text{nombre de mots de code complets}) + \mu^{\perp} \tag{3.22}$$

tel que le nombre de mots de code complets est égal à :

$$\frac{l - (a_{l,i} + b_{l,i})}{n} = \begin{cases} q & \text{si } r \geq a_{l,i} \\ q - 1 & \text{si } r < a_{l,i} \end{cases} \tag{3.23}$$

Alors, l'équation (3.22) peut s'écrire sous la forme :

$$s_{l,i} = \begin{cases} k \cdot q + \mu^{\perp} & \text{si } r \geq a_{l,i} \\ k \cdot (q - 1) + \mu^{\perp} & \text{si } r < a_{l,i} \end{cases} \tag{3.24}$$

Nombre de bits d'information $\text{beg}_{l,i}$ **et** $\text{end}_{l,i}$

Avec un mot de code, il est possible de récupérer au maximum k bits d'information. Donc, avec les $a_{l,i}$ bits de début et $b_{l,i}$ bits de fin, il sera possible de récupérer :

$$\begin{aligned} \text{beg}_{l,i} &= \min(k, a_{l,i}) = \begin{cases} a_{l,i} & \text{si } a_{l,i} \leq k \\ k & \text{si } a_{l,i} > k \end{cases} \\ \text{end}_{l,i} &= \min(k, b_{l,i}) \end{aligned} \tag{3.25}$$

En se basant sur le principe de raisonnement par récurrence, nous démontrons dans l'annexe C.2 que :

$$a_{l,i} = b_{l,\beta-1-i}, \forall i \in [\![0, \beta - 1]\!] \tag{3.26}$$

On peut déduire à partir de cette relation que les bits du bloc $b_{l,i}$ permettent de récupérer le même nombre de bits d'information que les bits du bloc $a_{l,i}$, ce qui permet d'obtenir :

$$\sum_{i=0}^{\beta-1} \mathrm{beg}_{l,i} = \sum_{i=0}^{\beta-1} \mathrm{end}_{l,i} \tag{3.27}$$

Calculons maintenant la somme de l'équation (3.27). Si on considère $l = \gamma \cdot n + r$ et $l' = \delta \cdot n + r'$, où γ et δ correspondent respectivement aux quotients de la division de l par n et l' par n avec $r' > r$ et $\mathrm{pgcd}(r', n) = \mathrm{pgcd}(r, n) = d$, on peut démontrer que pour $i \in [\![0, \beta - 1]\!]$, il existe un unique $j \in [\![0, \beta - 1]\!]$ où $a_{l,i} = a_{l',j}$. Les détails de cette démonstration sont exposés dans l'annexe C.3. Ainsi, si on considère $r = u \cdot d$ avec $u \in [\![0, \beta - 1]\!]$, la somme $\sum_{i=0}^{\beta-1} \mathrm{beg}_{l,i}$ est indépendante de u. Donc, pour simplifier le calcul, on prend $r = (\beta - 1) \cdot d$. Dans ce cas, on peut démontrer à travers le raisonnement par récurrence que $a_{l,i} = i \cdot d$ (voir annexe C.3). Par conséquent, avec les $i \cdot d$ bits de début de la matrice $\mathbf{R}_{l,i}$, il sera possible de récupérer :

$$\mathrm{beg}_{l,i} = \min(k, i \cdot d) = \begin{cases} i \cdot d & \text{si } i \cdot d \leq k \Rightarrow 0 \leq i \leq \left\lfloor \frac{k}{d} \right\rfloor \\ k & \text{si } \left\lfloor \frac{k}{d} \right\rfloor < i < \beta \end{cases} \tag{3.28}$$

Or, on a $\mathrm{beg}_{l,0} = 0$. Alors, la somme (3.27) est déterminée par :

$$\sum_{i=0}^{\beta-1} \mathrm{beg}_{l,i} = \sum_{i=1}^{\left\lfloor \frac{k}{d} \right\rfloor} i \cdot d + \sum_{i=\left\lfloor \frac{k}{d} \right\rfloor + 1}^{\beta-1} k \tag{3.29}$$

En utilisant la formule :

$$\sum_{i=1}^{\alpha} i = \frac{\alpha \cdot (\alpha + 1)}{2}, \tag{3.30}$$

on obtient :

$$\sum_{i=0}^{\beta-1} \mathrm{beg}_{l,i} = d \cdot \frac{\left\lfloor \frac{k}{d} \right\rfloor \cdot \left(\left\lfloor \frac{k}{d} \right\rfloor + 1 \right)}{2} + k \cdot \max\left(0, \beta - 1 - \left\lfloor \frac{k}{d} \right\rfloor\right) \tag{3.31}$$

Le rang global de $\mathbf{R}_l$

En remplaçant, dans l'équation (3.24), $a_{l,i}$ par $i \cdot d$, la somme de $s_{l,i}$ pour $i \in [\![0, \beta - 1]\!]$ sera égale à :

$$\sum_{i=0}^{\beta-1} s_{l,i} = k \cdot \left(\frac{l - n}{d} + 1 \right) + \beta \cdot \mu^{\perp} \tag{3.32}$$

Ainsi, en utilisant les équations (3.31) et (3.32), la simplification de $\varphi(l)$ correspondant à la somme du rang des matrices $\mathbf{R}_{l,i}$ est donnée par :

$$\varphi(l) = \sum_{i=0}^{\beta-1} \mathrm{rang}(\mathbf{R}_{l,i}) = \sum_{i=0}^{\beta-1} s_{l,i} + 2 \cdot \sum_{i=0}^{\beta-1} \mathrm{beg}_{l,i} \tag{3.33}$$

En utilisant l'équation (3.33), il est possible de vérifier que le rang des matrices $\mathbf{R}_{\alpha \cdot n}$, pour $\alpha \cdot n \geq n_a$, est égal à :

$$\mathbf{R}_{\alpha \cdot n} = \varphi(\alpha \cdot n) = \frac{k}{n} \cdot l + \mu^{\perp} \tag{3.34}$$

Le lecteur intéressé pourra se référer à l'annexe C.4 où une démonstration détaillée de l'équation (3.34) est exposée.

Exemple 3.30.

Pour illustrer le phénomène des chutes de rang multiples, nous prenons l'exemple du $\mathcal{C}(6,2,3)$ code convolutif, la matrice de codage utilisée dans cet exemple étant composée de trois sous-matrices :

$$\mathbf{F}_0 = \begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \end{pmatrix}, \mathbf{F}_1 = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}, \mathbf{F}_2 = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 \end{pmatrix}$$

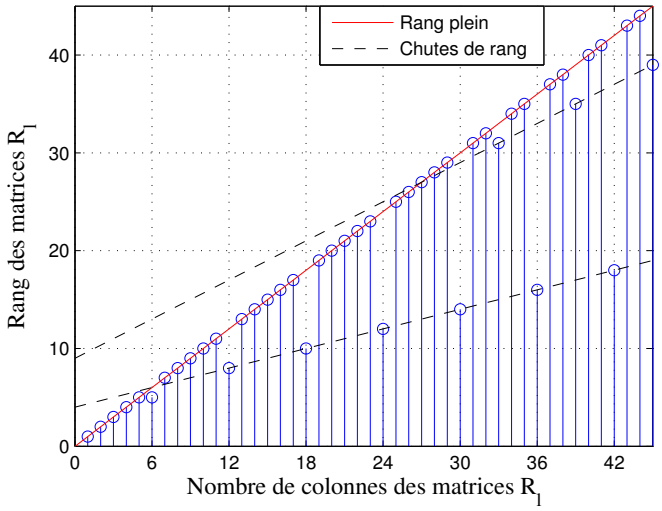


Figure 3.6 — Rang des matrices $\mathbf{R}_l$ pour le code convolutif $\mathcal{C}(6,2,3)$

Le nombre de colonnes l des matrices $\mathbf{R}_l$ varie entre 1 et 45. Le comportement du rang de ces matrices est représenté sur la figure 3.6. Les deux droites en tirées décrivent le comportement des chutes de rang des matrices $\mathbf{R}_l$. L'existence de ces droites multiples est liée aux paramètres du code. Les expressions qui permettent de déterminer les valeurs du rang de $\mathbf{R}_l$ sont données par les équations (3.18) et (3.33). En effet, si on prend $l_1 = 18$ et $l_2 = 39$, les rangs des matrices $\mathbf{R}_{l_1}$ et $\mathbf{R}_{l_2}$ sont décrits ci-dessous.

- Pour $l_1 = 18$, les paramètres requis pour calculer le rang de $\mathbf{R}_{l_1}$ sont :

$$\begin{cases} d = \text{pgcd}(l_1, n) = \text{pgcd}(18, 6) = 6 \\ \beta = \dfrac{n}{d} = \dfrac{6}{6} = 1 \end{cases}$$

En utilisant les équations (3.31) et (3.32), les sommes $\sum_{i=0}^{\beta-1} \text{beg}_{l_1,i}$ et $\sum_{i=0}^{\beta-1} s_{l_1,i}$ sont égales à :

$$\begin{cases} \sum_{i=0}^{\beta-1} \text{beg}_{l_1,i} = 6 \cdot \dfrac{\frac{2}{6} \cdot \left(\frac{2}{6} + 1\right)}{2} + 2 \cdot \max\left(0, 1 - 1 - \dfrac{2}{6}\right) = 0 \\ \sum_{i=0}^{\beta-1} s_{l_1,i} = 2 \cdot \left(\dfrac{18-6}{6} + 1\right) + 1 \cdot 4 = 10 \end{cases}$$

En utilisant l'équation (3.33), le rang de la matrice $\mathbf{R}_{l_1}$ est :

$$\text{rang}(\mathbf{R}_{l_1}) = \sum_{i=0}^{\beta-1} s_{l_1,i} + 2 \cdot \sum_{i=0}^{\beta-1} \text{beg}_{l_1,i} = 10 + 2 \cdot 0 = 10$$

Comme $l_1 = 3 \cdot n > n_a$, on peut appliquer l'expression (3.34) pour calculer le rang des

matrices $\mathbf{R}_{\alpha \cdot n}$:

$$\text{rang}(\mathbf{R}_{3 \cdot n}) = \varphi(\mathbf{R}_{3 \cdot n}) = \frac{2}{6} \cdot 18 + 4 = 10$$

On peut vérifier que nous trouvons la même valeur.

- Pour $l_2 = 39$, les paramètres requis pour calculer le rang de $\mathbf{R}_{l_2}$ sont :

$$\begin{cases} d = \text{pgcd}(l_2, n) = \text{pgcd}(39, 6) = 3 \\ \beta = \dfrac{n}{d} = \dfrac{6}{3} = 2 \end{cases}$$

Les sommes $\sum_{i=0}^{\beta-1} \text{beg}_{l_2,i}$ et $\sum_{i=0}^{\beta-1} s_{l_2,i}$ sont égales à :

$$\begin{cases} \sum_{i=0}^{\beta-1} \text{beg}_{l_2,i} = 3 \cdot \dfrac{\frac{2}{3} \cdot \left(\frac{2}{3} + 1\right)}{2} + 2 \cdot \max\left(0, 2 - 1 - \dfrac{2}{3}\right) = 2 \\ \sum_{i=0}^{\beta-1} s_{l_2,i} = 2 \cdot \left(\dfrac{39-6}{3} + 1\right) + 2 \cdot 4 = 32 \end{cases}$$

Alors, le rang de la matrice $\mathbf{R}_{l_2}$ est :

$$\text{rang}(\mathbf{R}_{l_2}) = \sum_{i=0}^{\beta-1} s_{l_2,i} + 2 \cdot \sum_{i=0}^{\beta-1} \text{beg}_{l_2,i} = 32 + 2 \cdot 2 = 36$$

On peut observer sur la figure 3.6 que la matrice de taille $l = 6$ qui doit être de rang plein présente une chute de rang de 1. Comme mentionné dans la sous-section 3.3.2, ce comportement est justifié par l'existence d'une combinaison linéaire entre les colonnes de la matrice de codage. Même s'il existe des chutes de rang multiples, on peut toujours identifier les paramètres du code à travers la droite des chutes de rang maximales.

3.4 Détection et identification des paramètres des codes non-binaires

Nous présentons dans cette partie le principe de l'algorithme d'identification aveugle des paramètres d'un code non-binaire. La plupart des travaux se sont limités à l'identification des codes binaires en se basant sur le calcul du rang des matrices construites à partir des données reçues. Dans cette section, la méthode du rang présentée dans la section 3.2 est adaptée aux codes correcteurs d'erreurs dans GF(2^m) lorsque les données reçues sont synchronisées et non entachées d'erreurs, ceci afin d'évaluer la faisabilité d'une telle méthode d'identification avant d'envisager dans le chapitre 4 le cas où il y a des erreurs. Notons que l'algorithme généralisé aux codes non-binaires dans le cas d'une transmission parfaite a fait l'objet d'une publication dans la conférence ICCCN 2011 [ZMGR11].

3.4.1 Principe de l'algorithme d'identification

Dans le cas des codes non-binaires, les matrices $\mathbf{R}_l$ définies dans la sous-section 3.2.1 sont construites à partir des données non-binaires. De ce fait, le rang pour chaque matrice $\mathbf{R}_l$ est calculé dans GF(2^m). La différence entre la construction des codes non-binaires et celle des codes binaires réside dans la génération des symboles non-binaires qui sont des éléments du corps GF(2^m). Pour cette raison, la connaissance des paramètres du corps de Galois (GF(2^m)) utilisés à l'émission est généralement requise à la réception pour identifier en aveugle les paramètres du code et effectuer

par la suite l'opération de décodage. Par ailleurs, si les paramètres du corps de Galois utilisés à l'émission (i.e le paramètre m et le polynôme primitif $p(x)$) sont connus, l'algorithme d'identification basé sur le critère du rang donne de bons résultats puisqu'il permet d'identifier l'ensemble des paramètres du code. Afin de mettre en évidence l'impact des paramètres du corps de Galois sur l'algorithme d'identification, une étude des comportements du rang sur les codes Reed-Solomon, les codes convolutifs dans GF(2^m) et les codes LDPC dans GF(2^m) est présentée dans la section 3.4.2.

3.4.2 Impact des paramètres du corps de Galois

Dans cette section, nous étudions l'effet du paramètre m et du polynôme primitif $p(x)$ sur l'algorithme du rang. Nous avons adapté la méthode d'identification des paramètres des codes dans GF(2^m) sous différentes hypothèses de bonne ou de mauvaise identification des paramètres du corps, c'est-à-dire le paramètre m et le polynôme primitif réellement utilisés à l'émission. Nous noterons $\tilde{m}$ le paramètre utilisé pour calculer le rang des matrices $\mathbf{R}_l$ et $\tilde{p}(x)$ le polynôme primitif du corps GF($2^{\tilde{m}}$) utilisé pour obtenir les symboles.

1. Si $\tilde{m} = m$ et $\tilde{p}(x) = p(x)$

Dans ce cas, la méthode décrite dans la section 3.2.1 peut être utilisée afin d'identifier les paramètres du code non-binaire.

Pour les autres hypothèses, avant d'appliquer l'algorithme d'identification, il sera nécessaire de convertir les données reçues en des symboles appartenant au corps de Galois GF($2^{\tilde{m}}$), où ce corps est supposé correspondre au corps de Galois utilisé par l'émetteur. Dans ce cas, les matrices $\mathbf{R}_l$ sont construites à partir des nouveaux symboles dans GF($2^{\tilde{m}}$).

2. Si $\tilde{m} = 1$ (GF(2))

Dans ce cas, les données reçues sont transformées en des symboles binaires et le rang est calculé dans le corps GF(2). L'algorithme d'identification basé sur le critère du rang nous permet d'estimer les paramètres du code équivalent dans GF(2). En effet, le rang des matrices $\mathbf{R}_l$ présente deux comportements différents. Dans cette configuration, les paramètres sont estimés à un facteur multiplicatif près qui est m : $\tilde{k} = m \cdot k$, $\tilde{n} = m \cdot n$ et $\tilde{\mu}^{\perp} = m \cdot \mu^{\perp}$. En réalité, ce sont les paramètres du code équivalent dans GF(2) puisque tous les corps GF(2^m) sont des extensions de GF(2).

3. Si $\tilde{m} = m$ et $\tilde{p}(x) \neq p(x)$

Le polynôme primitif s'avère en pratique indispensable pour générer les éléments d'un corps fini comme illustré dans le chapitre 1. Nous avons testé cette configuration sur différents codes non-binaires. Pour chaque code les résultats obtenus sont différents. En effet, le changement du polynôme primitif provoque soit des rangs pleins ou des chutes de rang de 1 pour quelques matrices. Ce phénomène sera examiné et analysé en détail dans le chapitre 6. Dans le cas où toutes les matrices $\mathbf{R}_l$ présentent des rangs pleins, on ne peut pas détecter la présence du code et identifier ses paramètres. Par contre, l'existence des chutes de 1 dans quelques matrices indique la présence d'un code mais ne permet pas d'identifier une liste exhaustive de ses paramètres.

4. Si $\tilde{m} \neq 1$ et $\tilde{m} \neq m$

Dans ce cas, les tests effectués en fonction du rendement du code peuvent nous donner deux résultats différents. Dans un premier cas, toutes les matrices sont de rang plein, donc aucun code ne peut être détecté. Dans un second cas, quelques matrices présentent des chutes de rang. Ce dernier cas est en cours d'étude afin de trouver une explication théorique.

Pour illustrer l'impact des paramètres du corps de Galois sur notre méthode d'identification, nous avons adapté la méthode aux codes Reed-Solomon, LDPC et convolutifs construits dans GF(2^m). Afin de comparer les comportements de chaque code, nous avons normalisé le rang des matrices pour chaque valeur de $\tilde{m}$ avec un rapport $\tilde{m}/m$ pour superposer les différentes chutes de rang des codes équivalents.

Notons que le logiciel utilisé pour nos simulations est Matlab version 2010 qui contient une bibliothèque comportant des fonctions sur les corps de Galois.

3.4.2.1 Codes convolutifs dans GF(2^m)

Dans cette partie, nous illustrons l'impact des paramètres du corps de Galois sur l'identification des paramètres de trois codes convolutifs non-binaires : code convolutif $\mathcal{C}(2,1,3)$ dans GF(2^4), code convolutif $\mathcal{C}(3,2,3)$ dans GF(2^2) et code convolutif $\mathcal{C}(3,1,3)$ dans GF(2^4). Pour étudier ces codes, les matrices génératrices considérées ont une taille $(n \times (k \cdot K))$ et s'écrivent sous la forme :

$$\mathbf{G} = \begin{pmatrix} \mathbf{F}_0^T & \mathbf{F}_1^T & \cdots & \mathbf{F}_{K-1}^T \end{pmatrix} \tag{3.35}$$

tel que chaque élément $\mathbf{F}_l, \forall l \in 0, \cdots, K-1$, est une sous-matrice non-binaire de taille $(k \times n)$, définie dans le deuxième chapitre.

- Code convolutif $\mathcal{C}(2,1,3)$ dans GF(2^4)

Pour étudier ce code, nous avons considéré la matrice génératrice utilisée dans [RW87] qui s'écrit sous la forme :

$$\mathbf{G} = \begin{pmatrix} 1 & 1 & \alpha^4 \\ 1 & \alpha & \alpha^4 \end{pmatrix} \tag{3.36}$$

Cinq cas sont représentés sur la figure 3.7 :

- Lorsque le bon corps et le bon polynôme primitif sont utilisés, le rang de la matrice $\mathbf{R}_l$ possède deux comportements. A partir des chutes de rang, les bons paramètres du code $\mathcal{C}(2,1,3)$ peuvent être estimés : $\tilde{n} = n = 2$, $\tilde{k} = k = 1$ et $\tilde{\mu}^{\perp} = \mu^{\perp} = 2 \Rightarrow K = 3$.

- Lorsque le mauvais polynôme primitif ou le corps de Galois GF(2^3) ou GF(2^2) est utilisé pour identifier les paramètres du code, toutes les matrices $\mathbf{R}_l$ sont de rang plein et il est impossible d'estimer les paramètres du code par cette méthode.

- L'utilisation du corps GF(2) à la réception permet d'avoir deux comportements du rang. Dans ce cas, les paramètres sont identifiés à un facteur multiplicatif $m = 4$: $\tilde{n} = 8 = m \cdot n$, $\tilde{k} = 4 = m \cdot k$ et $\tilde{\mu}^{\perp} = 8 = m \cdot \mu^{\perp}$.

Le tableau 3.1 présente un récapitulatif des paramètres identifiés du code $\mathcal{C}(2,1,3)$ dans GF(2^4) en fonction des hypothèses sur la bonne et la mauvaise estimation des paramètres du corps de Galois utilisé à l'émission.

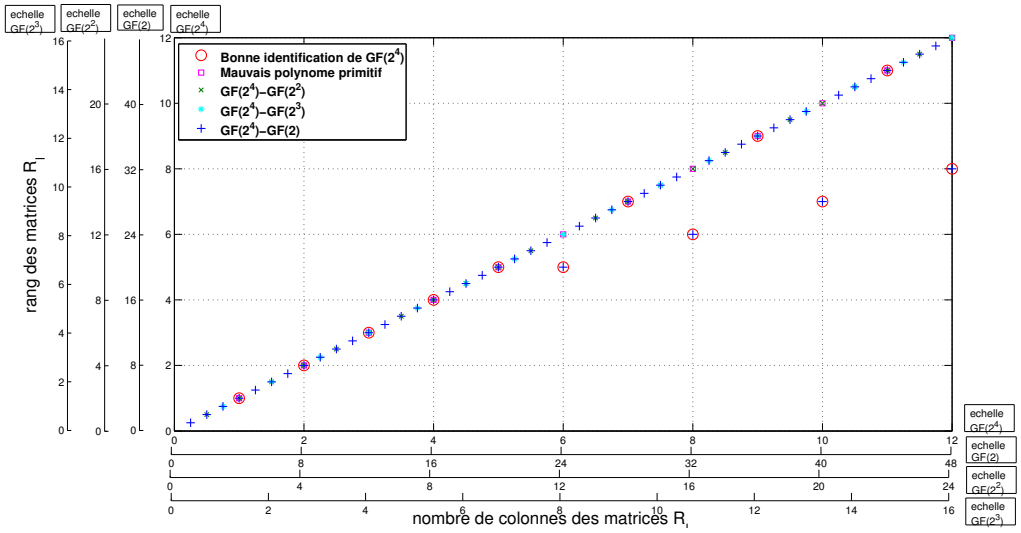


Figure 3.7 — Rang des matrices $\mathbf{R}_l$ pour le code convolutif $\mathcal{C}(2,1,3)$ dans GF(2^4)

Hypothèses	**Paramètres identifiés**
GF(2^4) et $p(x)$	$\tilde{n}=2$, $\tilde{k}=1$, $\tilde{\mu}^{\perp}=2$
GF(2)	code équivalent : $\tilde{n}=8$, $\tilde{k}=4$, $\tilde{\mu}^{\perp}=8$
Mauvais polynôme primitif	Aucun code détecté
GF(2^2)	
GF(2^3)	

Tableau 3.1 — Paramètres identifiés pour le code convolutif $\mathcal{C}(2,1,3)$ - GF(2^4) - $p(x) = x^4+x+1$.

Nous précisions que le temps consacré pour calculer le rang dans le cas de GF(2) est plus significatif que celui pris pour le calcul dans le cas de bons paramètres du corps.

- Code convolutif $\mathcal{C}(3,2,3)$ dans GF(2^2)

Pour étudier ce code, nous avons utilisé la matrice génératrice :

$$\mathbf{G} = \begin{pmatrix} 1 & 1 & \alpha^2 & \alpha & \alpha & \alpha^2 \\ 1 & \alpha & \alpha & \alpha & \alpha^2 & \alpha \\ 1 & \alpha^2 & 1 & \alpha & \alpha & 1 \end{pmatrix} \tag{3.37}$$

Trois cas sont représentés sur la figure 3.8 :

- Lorsque le bon corps et le bon polynôme primitif sont utilisés, le rang de la matrice $\mathbf{R}_l$ possède deux comportements et les bons paramètres du code $\mathcal{C}(3,2,3)$ peuvent être estimés : $\tilde{n}=n=3$, $\tilde{k}=k=2$ et $\tilde{\mu}^{\perp}=\mu^{\perp}=4 \Rightarrow \tilde{K}=K=3$.
- Lorsque le mauvais polynôme primitif ou le corps de Galois GF(2^4) est utilisé pour identifier les paramètres du code, toutes les matrices $\mathbf{R}_l$ sont de rang plein et il est impossible d'estimer les paramètres du code par cette méthode.
- Le rang calculé dans GF(2) possède deux comportements ce qui nous permet d'estimer les paramètres du code à un facteur multiplicatif $m=2$: $\tilde{n}=6=m\cdot n$, $\tilde{k}=4=m\cdot k$ et $\tilde{\mu}^{\perp}=8=m\cdot\mu^{\perp}$.

Le tableau 3.2 présente un récapitulatif des paramètres identifiés du code $\mathcal{C}(3,2,3)$ dans GF(2^2) en fonction des hypothèses sur la bonne ou la mauvaise estimation des paramètres du corps de Galois utilisé à l'émission.

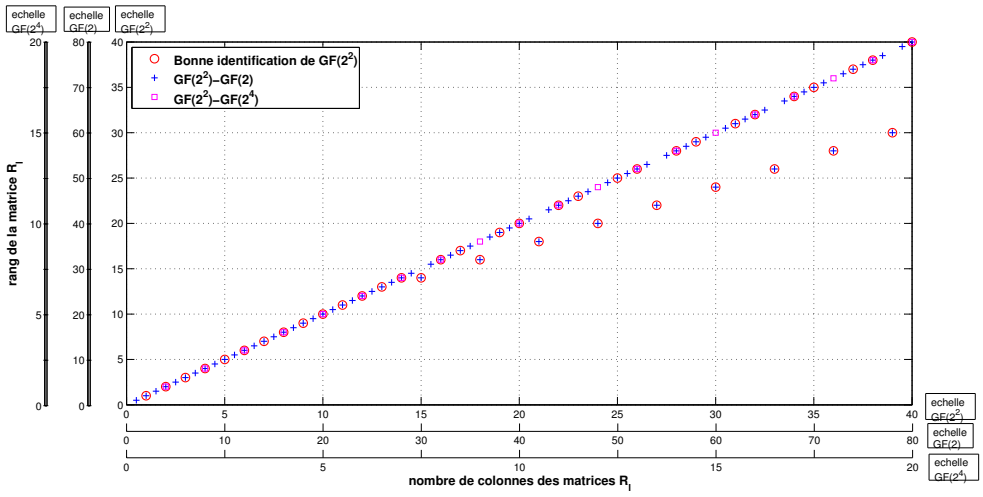


Figure 3.8 — Rang des matrices $\mathbf{R}_l$ pour le code convolutif $\mathcal{C}(3,2,3)$ dans GF(2^2)

Hypothèses	**Paramètres identifiés**
GF(2^2) et $p(x)$	$\tilde{n}=3$, $\tilde{k}=2$, $\tilde{\mu}^{\perp}=4$
GF(2)	code équivalent : $\tilde{n}=6$, $\tilde{k}=4$, $\tilde{\mu}^{\perp}=8$
GF(2^4)	Aucun code détecté

Tableau 3.2 — Paramètres identifiés pour le code convolutif $\mathcal{C}(3,2,3)$ - GF(2^2) - $p(x)=x^2+x+1$.

- Code convolutif $\mathcal{C}(3,1,3)$ dans GF(2^4)

Pour étudier ce code, nous avons considéré la matrice génératrice utilisée dans [RW87] qui s'écrit sous la forme :

$$\mathbf{G}=\begin{pmatrix}1 & 1 & \alpha^4\\ 1 & \alpha & \alpha^4\\ 1 & \alpha^2 & \alpha^9\end{pmatrix} \tag{3.38}$$

Cinq cas sont représentés sur la figure 3.11 :

- Dans le cas où le bon corps et le bon polynôme primitif sont utilisés, les rangs des matrices $\mathbf{R}_l$ possèdent deux comportements et les bons paramètres du code $\mathcal{C}(3,1,3)$ peuvent être estimés : $\tilde{n}=n=3$, $\tilde{k}=k=1$ et $\tilde{\mu}^{\perp}=\mu^{\perp}=2 \Rightarrow \tilde{K}=K=3$.

- Lorsque le mauvais polynôme primitif ou le corps de Galois GF(2^3) est utilisé pour identifier les paramètres du code, toutes les matrices $\mathbf{R}_l$ ont des rangs pleins et il est impossible d'estimer les paramètres du code.

- Si le corps GF(2) est utilisé pour construire les matrices $\mathbf{R}_l$ et calculer leurs rangs, les rangs calculés sont caractérisés par deux comportements. Pour ce code, il existe des chutes de rang maximales permettant d'identifier les paramètres du code équivalent dans GF(2) : $\tilde{n}=12=m\cdot n$, $\tilde{k}=4=m\cdot k$ et $\tilde{\mu}^{\perp}=8=m\cdot\mu^{\perp}$ avec $m=4$, mais il existe également certaines matrices qui présentent des chutes de rang moyennes. Ces dernières chutes de rang permettent

d'identifier les paramètres suivants : $\tilde{n} = 12$, $\tilde{k} = 8$ et $\tilde{\mu}^{\perp} = 20$. Ce dernier comportement peut être justifié par l'existence des combinaisons linéaires entre les colonnes de la matrice du codage du code dans GF(2).

- Lorsque le corps GF(2^2) est supposé identifié par le récepteur, il existe quelques matrices qui présentent des chutes de rang. Dans ce cas, le code est détecté et les paramètres estimés sont : $\tilde{n} = 6 = m \cdot n$, $\tilde{k} = 4$ et $\tilde{\mu}^{\perp} = 4$. Ces paramètres ne sont pas les bons paramètres du code $\mathcal{C}(3, 1, 3)$ dans GF(2^4). L'apparition de ce cas particulier peut être liée également à l'existence des combinaisons linéaires entre les colonnes de la matrice du codage construite dans GF(2^2).

Le tableau 3.3 présente un récapitulatif des paramètres identifiés du code $\mathcal{C}(3, 1, 3)$ dans GF(2^4) en fonction des hypothèses sur la bonne et la mauvaise estimation des paramètres du corps de Galois utilisé à l'émission.

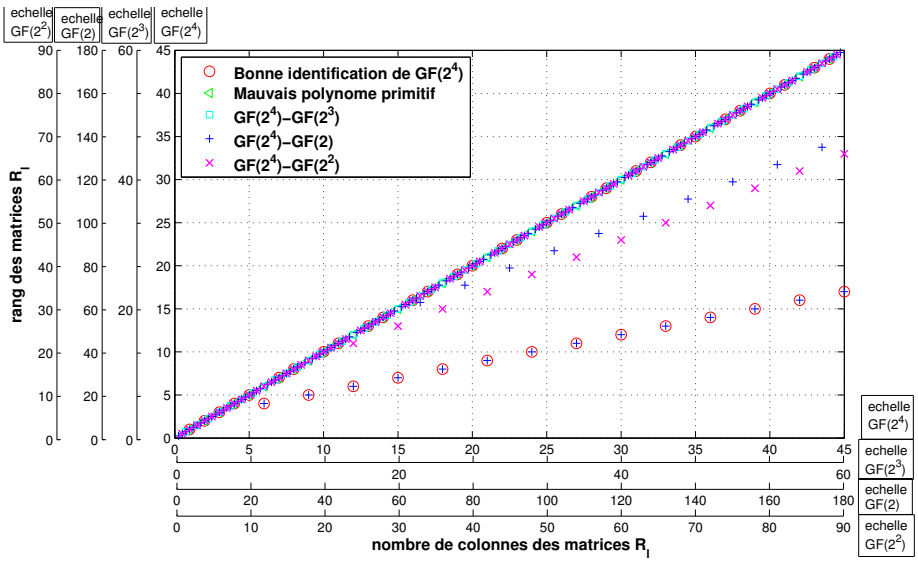


Figure 3.9 — Rang des matrices $\mathbf{R}_l$ pour le code convolutif $\mathcal{C}(3, 1, 3)$ dans GF(2^4)

Hypothèses	**Paramètres identifiés**
GF(2^4) et $p(x)$	$\tilde{n} = 3$, $\tilde{k} = 1$, $\tilde{\mu}^{\perp} = 2$
GF(2)	code équivalent : $\tilde{n} = 12$, $\tilde{k} = 4$, $\tilde{\mu}^{\perp} = 8$
	$\tilde{n} = 12$, $\tilde{k} = 8$, $\tilde{\mu}^{\perp} = 20$
GF(2^2)	$\tilde{n} = 6$, $\tilde{k} = 4$, $\tilde{\mu}^{\perp} = 4$
Mauvais polynôme primitif	Aucun code détecté
GF(2^3)	

Tableau 3.3 — Paramètres identifiés pour le code convolutif $\mathcal{C}(3, 1, 3)$ - GF(2^4) - $p(x) = x^4 + x + 1$.

3.4.2.2 Codes LDPC dans GF(2^m)

Nous présentons dans ce paragraphe l'impact des paramètres du corps de Galois sur l'identification des codes LDPC non-binaires. Nous considérons un exemple du code LDPC($n = 12$, $k = 6$)

dans GF(2^3) défini par la matrice de contrôle de parité :

$$\mathbf{H}_1 = \begin{pmatrix} 0 & 0 & 0 & 4 & 6 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 7 \\ 0 & 0 & 6 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 3 & 0 & 5 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 6 & 1 & 6 & 0 & 6 \\ 0 & 4 & 2 & 0 & 0 & 3 & 2 & 0 & 0 & 0 & 7 & 0 \end{pmatrix} \tag{3.39}$$

En fonction des hypothèses sur la bonne et la mauvaise identification des paramètres du corps GF(2^3), le rang des matrices $\mathbf{R}_l$ est représenté sur la figure 3.10.

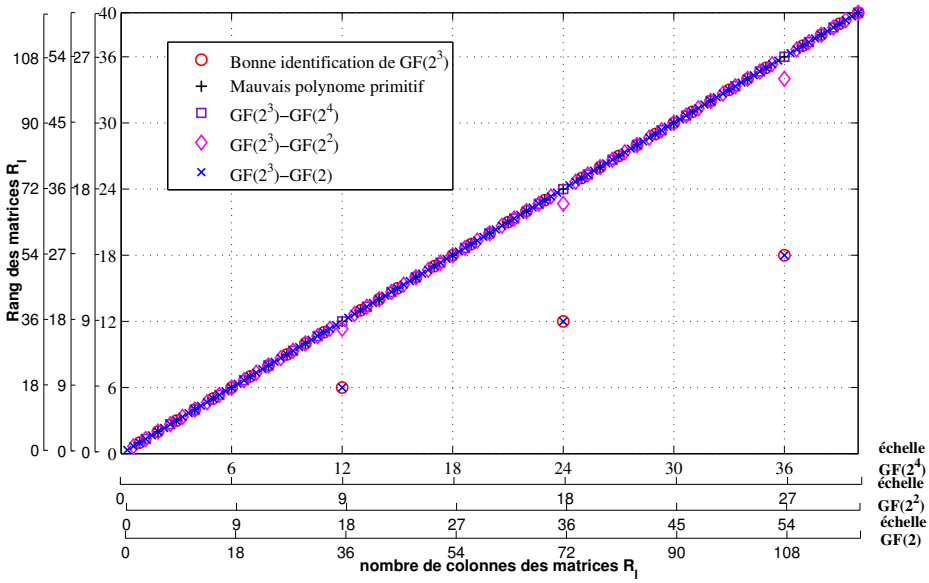


Figure 3.10 — Rang des matrices $\mathbf{R}_l$ pour le code LDPC(12, 6) dans GF(2^3) défini par $\mathbf{H}_1$

Dans le cas où les bons paramètres du corps de Galois sont utilisés, les bons paramètres du code sont identifiés. D'après la figure 3.10, on peut remarquer que l'utilisation du mauvais polynôme ou du corps GF(2^4) ne permet pas de détecter l'existence du code puisque toutes les matrices $\mathbf{R}_l$ sont de rang plein. Si le corps GF(2^2) est utilisé, on peut détecter l'existence du code, mais on ne peut pas identifier ses paramètres. Dans le cas de l'utilisation du corps GF(2) à la réception à la place du corps GF(2^3), les paramètres d'un code équivalent à un facteur multiplicatif près $m = 3$ sont identifiés.

Le tableau 3.4 présente un récapitulatif des paramètres identifiés du code LDPC(12,6) en fonction de la bonne et la mauvaise identification des paramètres du corps GF(2^3).

Hypothèses	**Paramètres identifiés**
GF(2^3) et $p(x)$	$\tilde{n} = 12$, $\tilde{k} = 6$
GF(2)	code équivalent : $\tilde{n} = 36$, $\tilde{k} = 24$
GF(2^2)	$\tilde{n} = 18$, $\tilde{k} = 12$
Mauvais polynôme primitif	Aucun code détecté
GF(2^4)	

Tableau 3.4 — Paramètres identifiés pour le code LDPC(12, 6) - GF(2^3) - $p(x) = x^3 + x + 1$.

Nous présentons un cas particulier de l'impact du mauvais polynôme primitif sur les comportements du rang pour le code LDPC(12, 6) défini par la matrice de contrôle de parité :

$$\mathbf{H}_2 = \begin{pmatrix} 5 & 0 & 2 & 7 & 2 & 7 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 1 \\ 0 & 0 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 7 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6 & 7 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 5 & 5 & 0 & 0 & 0 & 0 \\ 7 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 7 & 5 \end{pmatrix}$$

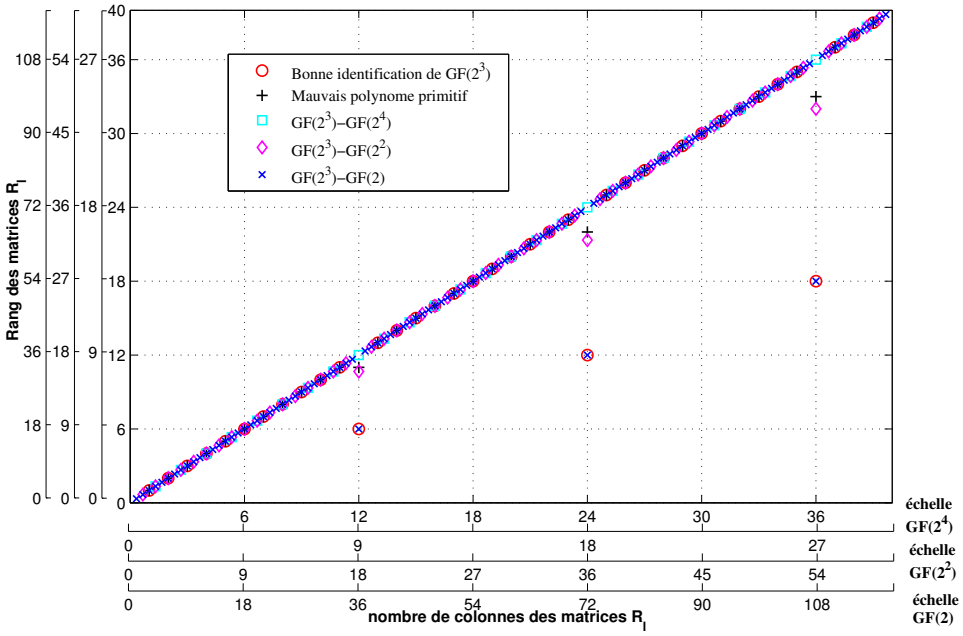


Figure 3.11 — Rang des matrices $\mathbf{R}_l$ pour le code LDPC(12, 6) dans GF(2^3) défini par $\mathbf{H}_2$

Les comportements du rang des matrices $\mathbf{R}_l$ sont représentés sur la figure 3.11 en fonction des hypothèses sur la bonne et la mauvaise identification des paramètres du corps GF(2^3). En comparant les comportements observés sur cette figure par rapport à ceux de la figure 3.10, nous pouvons remarquer des chutes de rang dans le cas de l'utilisation du mauvais polynôme primitif à la réception. L'existence de ces chutes sera justifiée théoriquement dans le chapitre 6. Mais, on peut remarquer à travers les deux derniers exemples de code LDPC(12,6) dans GF(2^3) que l'existence de ces chutes peut dépendre de la construction de la matrice de contrôle de parité du code.

3.4.2.3 Codes Reed-Solomon

Soit un code Reed-Solomon RS(15, 11) travaillant dans GF(2^4). Ce code est défini par le polynôme générateur :

$$g(x) = x^4 + 15 \cdot x^3 + 3 \cdot x^2 + x + 12 \tag{3.40}$$

Nous avons étudié le comportement du rang des matrices $\mathbf{R}_l$ en fonction de la bonne ou de la mauvaise identification des paramètres de GF(2^4) utilisé à l'émission. Quatre cas sont représentés sur la figure 3.12.

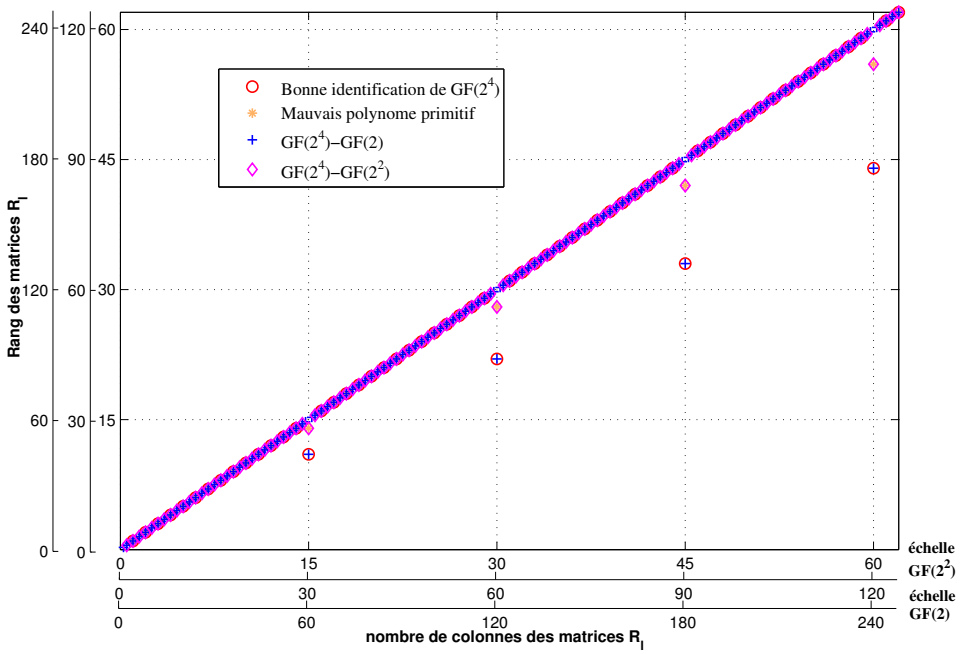


Figure 3.12 — Rang des matrices $\mathbf{R}_l$ pour le code RS$(15, 11)$ dans GF(2^4)

Dans le cas où le bon corps et le bon polynôme primitif sont utilisés, le rang des matrices $\mathbf{R}_l$ possède deux comportements et les bons paramètres du code Reed-Solomon estimés sont : $\tilde{n} = 15$ et $\tilde{k} = 11$. Par contre, lorsque le mauvais polynôme primitif est utilisé pour identifier les paramètres du code, certaines matrices $\mathbf{R}_l$ ont des rangs déficients et les paramètres estimés sont : $\tilde{n} = 15 = n$ et $\tilde{k} = 14 \neq k$. Dans ce cas, la bonne taille des mots de code est identifiée. Si le corps GF(2) est utilisé pour construire les matrices $\mathbf{R}_l$ et calculer leurs rangs, on remarque deux comportements. Dans ce cas, les paramètres identifiés sont $\tilde{n} = 60 = m \cdot n$ et $\tilde{k} = 44 = m \cdot k$ avec $m = 4$. Lorsque le corps GF(2^2) est supposé être identifié par le récepteur, il existe quelques matrices qui présentent des chutes de rang. Dans ce cas, le code est détecté et les paramètres estimés sont : $\tilde{n} = 30 = m \cdot n$ et $\tilde{k} = 28$. Ces paramètres ne sont pas les bons paramètres du code RS$(15, 11)$.

Le tableau 3.5 présente un récapitulatif des paramètres identifiés par notre algorithme en fonction des hypothèses sur la bonne et la mauvaise estimation des paramètres de GF(2^4).

Hypothèses	**Paramètres identifiés**
GF(2^4) et $p(x)$	$\tilde{n} = 15$, $\tilde{k} = 11$
GF(2)	code équivalent : $\tilde{n} = 60$, $\tilde{k} = 44$
Mauvais polynôme primitif	$n = 15$, $\tilde{k} = 14$
GF(2^2)	$\tilde{n} = 30$, $\tilde{k} = 28$

Tableau 3.5 — Paramètres identifiés pour le code RS$(15, 11)$ - GF(2^4) - $p(x) = x^4 + x + 1$.

3.5 Identification aveugle du code dual des codes non-binaires

Dans cette section, nous généralisons l'algorithme d'identification aveugle du code dual pour les codes convolutifs binaires présenté dans la thèse [Mar09] au cas des codes non-binaires. Nous verrons cet algorithme dans le cas des codes en bloc non-binaires, mais il peut être également appliqué aux codes convolutifs non-binaires.

Après avoir identifié les paramètres du code, nous construisons la matrice $\mathbf{R}_n$ de taille $(M \times n)$ dont le rang doit être inférieur ou égal à $n-1$. Pour identifier le code dual, il suffit de chercher une base de ce code qui comporte $n-k$ relations de parité permettant de construire une matrice de contrôle de parité $\mathbf{H}$. Cette matrice de parité et la matrice $\mathbf{R}_n$ doivent vérifier la relation :

$$\mathbf{R}_n \cdot \mathbf{H}^T = \mathbf{0} \tag{3.41}$$

Ainsi, la résolution du système (3.41) permet d'identifier une matrice de parité. Pour cela, il est nécessaire de disposer d'au moins n^2 symboles reçus. On peut remarquer à travers l'équation (3.41) que les relations de parité appartiennent au noyau de $\mathbf{R}_n$, noté $\ker(\mathbf{R}_n)$. De ce fait, le noyau $\ker(\mathbf{R}_n)$ est inclus dans l'espace du code dual.

L'idée pour résoudre notre problème est d'appliquer l'algorithme d'élimination de Gauss adapté aux corps finis pour obtenir une nouvelle matrice, notée $\mathbf{T}_n$, de taille $(M \times n)$. Un rappel sur le fonctionnement de cet algorithme est présenté dans l'annexe D. Cet algorithme permet de donner en sortie une matrice de taille $(n \times n)$, notée $\mathbf{A}_n$, qui décrit toutes les opérations de combinaisons de colonnes dans $\mathrm{GF}(2^m)$ réalisées sur la matrice $\mathbf{R}_n$ afin d'obtenir la matrice $\mathbf{T}_n$. Ainsi, la transformation effectuée sur la matrice $\mathbf{R}_n$ est décrite par une application linéaire sous la forme :

$$\mathbf{R}_n \cdot \mathbf{A}_n = \mathbf{T}_n \tag{3.42}$$

L'existence des colonnes nulles dans la matrice $\mathbf{T}_n$ indique la présence de colonnes dépendantes dans la matrice $\mathbf{R}_n$. La dépendance de ces colonnes a été engendrée par la redondance introduite par le code. Le nombre de colonnes dépendantes doit être inférieur ou égal à $n-k$. Notons $\mathbf{t}_i^{(n)}$ la i-ème colonne de la matrice $\mathbf{T}_n$ et $t_i^{(n)}(j)$ un élément de cette colonne appartenant à la j-ème ligne. La forme de la matrice $\mathbf{T}_n$ est décrite sur la figure 3.13.

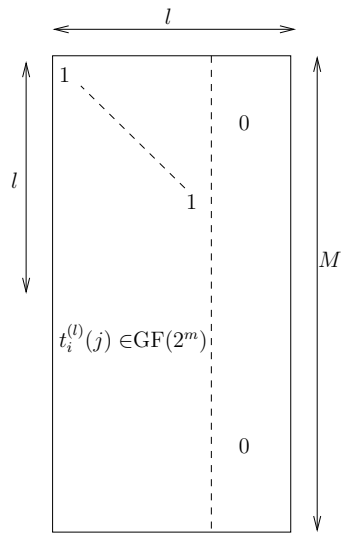


Figure 3.13 — Forme de la matrice $\mathbf{T}_n$ pour $l = n$

Dans le cas où $\mathbf{t}_i^{(n)}$ est une colonne nulle (i.e. $\mathbf{t}_i^{(n)} = \mathbf{0}$), nous aurons :

$$\mathbf{R}_n \cdot \mathbf{a}_i^{(n)} = \mathbf{0} \tag{3.43}$$

D'après cette équation, on peut déduire que la colonne $\mathbf{a}_i^{(n)}$ appartient au noyau de $\mathbf{R}_n$ dans lequel l'espace du code dual $\mathcal{C}^\perp$ est inclus ($\mathcal{C}^\perp \subset \ker(\mathbf{R}_n)$). Alors, cette colonne représente une relation de parité. Par conséquent, les relations de parité du code sont les colonnes de $\mathbf{A}_n$ qui appartiennent au noyau de $\mathbf{R}_n$. Notons $N_n(i)$ le nombre de zéros dans la colonne $\mathbf{t}_i^{(n)}$. Le principe de la recherche

d'une base de code dual est illustré par l'algorithme 1.

Algorithme 1: Identification d'une base du code dual dans le cas non bruité

Entrées : Les données reçues $\mathbf{r}$, la taille des mots de code n
Sorties : Une base du code dual $\mathcal{D}$

Construire $\mathbf{R}_n$ de taille $(M \times n)$ avec $\mathbf{r}$;
Triangulariser $\mathbf{R}_n \Longrightarrow \mathbf{T}_n = \mathbf{R}_n \cdot \mathbf{A}_n$;
pour $i = 1$ *à* n **faire**
  si $N_n(i) = M$ **alors**
    $\mathcal{D} \longleftarrow \mathcal{D} \cup \{\mathbf{a}_i^{(n)}\}$;
  fin
fin

Exemple 3.31.

Prenons l'exemple du code Reed-Solomon RS(7, 3) construit dans le corps de Galois GF(2^3). La matrice génératrice de ce code et la matrice de contrôle de parité sont telles que :

$$\mathbf{G} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 3 & 6 & 7 & 5 \\ 1 & 4 & 6 & 5 & 2 & 3 & 7 \end{pmatrix} \tag{3.44}$$

$$\mathbf{H} = \begin{pmatrix} 1 & 2 & 4 & 3 & 6 & 7 & 5 \\ 1 & 4 & 6 & 5 & 2 & 3 & 7 \\ 1 & 3 & 5 & 4 & 7 & 2 & 6 \\ 1 & 6 & 2 & 7 & 4 & 5 & 3 \end{pmatrix} \tag{3.45}$$

La construction de ce code est décrite dans la section 2.3.1.3. Nous supposons que les paramètres du code $n = 7$ et $k = 3$ sont connus à la réception. Notre objectif est d'identifier une base du code dual lorsque le train de symboles reçu $\mathbf{r}$ est synchronisé et non entaché d'erreurs :

$$\mathbf{r} = \begin{pmatrix} 7 & 3 & 7 & 5 & 5 & 3 & 1 & 0 & 1 & 7 & 6 & 1 & 6 & 0 & \cdots \end{pmatrix} \tag{3.46}$$

La matrice $\mathbf{R}_7$ de taille (7×7) construite à partir des symboles reçues $\mathbf{r}$ a la forme :

$$\mathbf{R}_7 = \begin{pmatrix} 7 & 3 & 7 & 5 & 5 & 3 & 1 \\ 0 & 1 & 7 & 6 & 1 & 6 & 0 \\ 4 & 2 & 5 & 0 & 1 & 3 & 7 \\ 2 & 0 & 6 & 2 & 6 & 4 & 4 \\ 4 & 4 & 1 & 2 & 7 & 1 & 7 \\ 5 & 5 & 1 & 4 & 0 & 1 & 0 \\ 2 & 3 & 3 & 0 & 1 & 2 & 0 \end{pmatrix}$$

L'algorithme du pivot de Gauss appliqué à la matrice $\mathbf{R}_7$ donne en sortie les matrices $\mathbf{T}_7$ et $\mathbf{A}_7$:

$$\mathbf{T}_7 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 6 & 3 & 1 & 0 & 0 & 0 & 0 \\ 3 & 5 & 7 & 0 & 0 & 0 & 0 \\ 6 & 5 & 1 & 0 & 0 & 0 & 0 \\ 2 & 3 & 2 & 0 & 0 & 0 & 0 \\ 3 & 6 & 3 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\mathbf{A}_7 = \begin{pmatrix} 4 & 7 & 7 & 3 & 2 & 1 & 3 \\ 0 & 1 & 4 & 5 & 5 & 1 & 4 \\ 0 & 0 & 6 & 7 & 6 & 1 & 6 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

D'après l'algorithme 1, en utilisant les matrices $\mathbf{T}_7$ et $\mathbf{A}_7$, on peut déterminer la base :

$$\mathcal{D} = \{\mathbf{a}_4^{(7)}, \mathbf{a}_5^{(7)}, \mathbf{a}_6^{(7)}, \mathbf{a}_7^{(7)}\}$$

Par conséquent, la matrice de contrôle de parité du code identifié peut s'écrire :

$$\mathbf{H}' = \begin{pmatrix} 3 & 5 & 7 & 1 & 0 & 0 & 0 \\ 2 & 5 & 6 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 3 & 4 & 6 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Cette matrice correspond à la matrice équivalente de la matrice $\mathbf{H}$ et elle vérifie la relation :

$$\mathbf{G} \cdot \mathbf{H}^T = \mathbf{0}$$

3.6 Conclusion

Dans ce chapitre, nous avons présenté une démonstration théorique et algébrique des comportements du rang qui sont exploités dans de nombreux travaux pour identifier en aveugle les paramètres des codes correcteurs d'erreurs binaires. Ensuite, nous avons mis en évidence l'impact de la matrice génératrice sur quelques chutes de rang. Notre étude a donné une explication à l'existence des chutes de rang multiples qui dépendent des codes ayant des paramètres spécifiques. Dans ce contexte, nous avons proposé une expression générale du calcul du rang afin de traiter les comportements étudiés du critère du rang.

Nous avons ensuite généralisé l'algorithme d'identification des paramètres des codes binaires basé sur le critère du rang au cas des codes correcteurs d'erreurs dans GF(2^m). Nous avons étudié l'influence de la mauvaise estimation des paramètres du corps de Galois sur l'identification des paramètres des codes convolutifs dans GF(2^m), codes LDPC dans GF(2^m) et codes de Reed-Solomon. Cette étude a démontré la pertinence de l'algorithme d'identification des codes dans GF(2^m) sous l'hypothèse que les paramètres des corps de Galois (m et le polynôme primitif) utilisés à l'émission soient connus ou correctement identifiés par le récepteur. En supposant que les paramètres du code non-binaire ont été identifiés, nous avons présenté dans la dernière partie de ce chapitre un algorithme pour identifier en aveugle une base du code dual dans des conditions de transmission parfaite.

Dans le prochain chapitre, nous nous attaquerons au problème plus ardu d'identification des paramètres des codes non-binaires lorsque les données codées sont entachées d'erreurs.

CHAPITRE 4

Identification aveugle des codes dans le cas bruité

4.1 Introduction

Dans ce chapitre, nous nous intéressons à l'identification aveugle de la taille des mots de codes et d'une base du code dual en considérant une transmission bruitée et des codes qui n'engendrent pas des chutes multiples. Les travaux sur la thématique de l'identification aveugle des paramètres du bloc de codage canal ont été limités pour la plupart à des codes binaires. Des techniques d'identification aveugle des codes en bloc ont été proposées dans [BG03, CF09]. Dans le chapitre précédent, une méthode basée sur le critère du rang a été proposée pour des codes binaires et non-binaires. Nous avons démontré la pertinence de cette méthode pour des codes dans $\mathrm{GF}(q)$ sous l'hypothèse que les paramètres du corps de Galois (l'entier m et le polynôme primitif) utilisés à l'émission soient connus ou correctement identifiés par le récepteur. Dans le cas d'une transmission bruitée, les auteurs dans [SH05, MGB11] ont proposé une technique pour les codes binaires basée sur la recherche des colonnes presque dépendantes dans les matrices construites à partir des bits reçues. Dans la première partie de ce chapitre, nous généralisons en premier lieu cette technique pour identifier la taille des codes non-binaires. Puis, nous proposons deux nouvelles techniques plus robustes qui ne nécessitent pas en entrée la connaissance de la probabilité d'erreur du canal. La deuxième partie de ce chapitre est consacrée à l'identification d'une base du code dual en utilisant tout d'abord une décision ferme ; puis nous montrons que l'utilisation conjointe d'une décision souple associée à une prise en compte des probabilités de fiabilité de l'estimation des relations de parité déjà identifiées permet d'améliorer la probabilité d'identification de $n-k$ relations de parité. Dans la dernière partie, une analyse de performances en termes de probabilités d'identification est exposée.

4.2 Identification aveugle de la taille des mots de code

Dans un contexte non-coopératif, le récepteur doit être capable d'identifier en aveugle les paramètres des codes correcteurs d'erreurs avec la seule connaissance des données reçues. L'objectif de cette section est de proposer une approche robuste permettant d'identifier en aveugle la taille des mots de code pour des codes non-binaires lorsque les données reçues sont entachées d'erreurs. Dans un tel contexte, la plupart des résultats de recherche publiés ont été limités jusqu'à présent à la reconnaissance aveugle de la taille des mots de code pour des codes binaires. Dans cette thèse, nous proposons trois approches pour identifier en aveugle la taille de mots de code pour des codes non-binaires. La première approche est une généralisation d'une méthode existante utilisée pour les codes binaires. De ce fait, nous présentons tout d'abord l'état de l'art des techniques existantes pour des codes binaires et décrivons la technique qui nous intéresse. Ensuite, nous généralisons cette technique au cas des codes non-binaires. A l'aide de cette généralisation, nous pouvons mettre en avant des points faibles de la technique existante. Ceci nous a permis de proposer deux nouvelles approches à l'aide d'approches probabilistes et des critères de la moyenne et de la variance.

4.2.1 Codes binaires

L'idée des techniques existantes pour identifier en aveugle la taille des mots de code pour des codes binaires est de trouver des relations de de parité appartenant à une base du code dual. Pour ce faire, une approche basée sur la recherche des mots de code de faible poids de Hamming [Ste89b, Can98b] a été introduite par Planquette [Pla96]. Cette approche a été améliorée tout d'abord par Valembois [Val01] en utilisant des tests d'hypothèses statistiques et par Cluzeau dans [Clu06a, CF09]. Une deuxième technique basée sur la théorie de l'algèbre linéaire a été introduite dans [BG03] pour un canal non-bruité. Avec cette technique, on peut identifier la taille des mots de code en étudiant le rang des matrices composées des bits reçus réarrangés de façon appropriée. Dans le cas d'une transmission bruitée, un algorithme d'élimination de Gauss a été appliqué dans [SH05, SHB09, MGB11] aux matrices composées des bits reçus entachés d'erreurs afin de trouver le nombre de colonnes presque dépendantes permettant l'identification de la taille des mots de code dans le cas des codes correcteurs d'erreurs binaires. Dans nos travaux, nous nous intéressons à cette technique que nous généraliserons au cas des codes non-binaires.

4.2.1.1 Principe d'identification

Dans le cadre de l'identification pour les codes binaires, nous avons considéré un canal binaire symétrique de probabilité d'erreur p_e. Les données reçues bruitées r_i, $\forall i \in \{1, ..., L\}$, sont réorganisées pour construire des matrices $\tilde{\mathbf{R}}_l$, de taille $(M \times l)$, avec $M > l$ (voir figure 3.1). Le nombre de colonnes l varie entre 2 et l_{max} et le nombre de lignes M dépendant de l est déterminé par $M = \lfloor L/l \rfloor$.

Dans le cas d'une transmission non-bruitée, les matrices $\tilde{\mathbf{R}}_l = \mathbf{R}_l$ présentent des déficiences de rang pour $l = \alpha \cdot n$ et $l \geq n_a$, $\forall \alpha \in \mathbb{N}$, avec n la taille des mots de code. En pratique, le rang d'une matrice est calculé en déterminant le nombre de colonnes qui sont linéairement dépendantes, ce qui permet d'en déduire le nombre de celles qui sont indépendantes. Dans le contexte d'une transmission bruitée, cette dépendance est perturbée par la présence de bits erronés. Par conséquent, toutes les matrices $\tilde{\mathbf{R}}_l$ peuvent être de rang plein. Dans un tel contexte, les auteurs dans [SH05, MGB11] ont proposé deux algorithmes pour identifier les paramètres de codes convolutifs binaires et les paramètres d'entrelaceurs en blocs binaires. L'idée de ces deux algorithmes consiste à chercher les colonnes qui sont "presque dépendantes" dans les matrices $\tilde{\mathbf{R}}_l$ en utilisant l'algorithme de Gauss dans GF(2). Dans ce cas, les matrices $\tilde{\mathbf{R}}_l$ seront transformées en des matrices $\tilde{\mathbf{T}}_l$ qui s'expriment sous la forme :

$$\tilde{\mathbf{R}}_l \cdot \mathbf{A}_l = \tilde{\mathbf{T}}_l \tag{4.1}$$

où $\mathbf{A}_l$ est une matrice inversible de taille $(l \times l)$ qui représente les permutations et les combinaisons de colonnes. La matrice triangulaire obtenue $\tilde{\mathbf{T}}_l$, est de taille $(M \times l)$. La méthode d'identification de la taille des mots de code binaire présentée dans [MGB11] est basée sur la recherche des colonnes presque dépendantes en étudiant le nombre de 1 (ou 0) dans les colonnes des matrices $\tilde{\mathbf{T}}_l$.

4.2.1.2 Description de la méthode

Notons $B_l(i)$ une variable contenant le nombre de 0 de la i-ème colonne de la matrice $\tilde{\mathbf{T}}_l$. La variable $B_l(i)$ est étudiée en fonction de l :

- Pour $l \neq \alpha \cdot n$ ou $l < n_a$ avec n_a la taille de la première matrice de rang déficient : les matrices $\tilde{\mathbf{R}}_l$ seront de rang plein. Dans ce cas, la variable $B_l(i)$ suivra la loi binomiale $\mathcal{B}(M, 1/2)$.
- Pour $l = \alpha \cdot n$ et $l \geqslant n_a$: la variable $B_l(i)$ aura deux comportements en fonction de i :
  - Si i est l'indice d'une colonne presque dépendante : $B_l(i)$ suivra la loi binomiale $\mathcal{B}(M, P_i)$, avec P_i la probabilité qu'un élément de la i-ème colonne de $\tilde{\mathbf{T}}_l$ soit nul.

– Si i n'est pas l'indice d'une colonne presque dépendante : $B_l(i)$ suivra la loi binomiale $\mathcal{B}(M, 1/2)$.

Valembois a démontré dans sa thèse [Val00] que la probabilité P_i d'avoir un coefficient nul, noté $\tilde{t}_i^{(l)}(j)$, dans la i-ème colonne de la matrice $\tilde{\mathbf{T}}_l$ peut s'écrire :

$$P_i = Pr\left[\tilde{t}_i^{(l)}(j) = 0\right] = \frac{1 + (1 - 2 \cdot p_e)^{w_H(\mathbf{a}_i^{(l)})}}{2} \tag{4.2}$$

où $w_H(\mathbf{a}_i^{(l)})$ est le poids de Hamming de la i-ème colonne de la matrice $\mathbf{A}_l$, notée $\mathbf{a}_i^{(l)}$.

Notons γ le seuil de décision et $Q(l)$ l'ensemble contenant les colonnes presque dépendantes de la matrice $\tilde{\mathbf{R}}_l$. L'appartenance de la i-ème colonne de $\tilde{\mathbf{R}}_l$, notée $\mathbf{r}_i^{(l)}$, à $Q(l)$ peut être déterminée au regard des deux comportements de $B_l(i)$ comme suit :

$$\begin{cases} \text{Si } B_l(i) > \frac{M}{2} \cdot \gamma \text{ alors } \mathbf{r}_i^{(l)} \in Q(l) \\ \text{Si } B_l(i) \leq \frac{M}{2} \cdot \gamma \text{ alors } \mathbf{r}_i^{(l)} \notin Q(l) \end{cases} \tag{4.3}$$

Les auteurs dans [MGB11] ont montré que le seuil optimal γ_{opt} dépend de la probabilité d'erreur du canal p_e. Pour une colonne i, le seuil optimal γ_{opt} a été calculé par la minimisation de la somme des probabilités de fausse alarme et de non-détection. Ce seuil est défini par :

$$\gamma_{opt} = \arg\min_{\gamma} \left(1 + \sum_{j=0}^{\lfloor \frac{M-l}{2} \cdot \gamma \rfloor} \binom{M-l}{j} \cdot \left[2^{l-M} - P_i^j \cdot (1 - P_i)^{M-l-j} \right] \right) \tag{4.4}$$

Après avoir déterminé le seuil optimal pour chaque colonne i, on peut détecter la colonne qui appartient à l'ensemble $Q(l)$. La taille des mots de code peut être identifiée par :

$$n = \text{mode}(\text{diff}(\mathcal{I})) \tag{4.5}$$

avec $\mathcal{I}$ l'ensemble défini par :

$$\mathcal{I} = \{l \in [\![2, l_{max}]\!] | card(Q(l)) \neq 0\} \tag{4.6}$$

où les fonctions diff($\mathbf{x}$) et mode($\mathbf{x}$) sont définies par :

- Fonction diff($\mathbf{x}$) : la sortie de cette fonction est un vecteur de taille $s - 1$ dont les éléments sont calculés comme la différence entre deux éléments successifs du vecteur $\mathbf{x} = \begin{pmatrix} x(1) & x(2) & \cdots & x(s) \end{pmatrix}$:

$$\text{diff}(\mathbf{x}) = \begin{pmatrix} x(2) - x(1) & \cdots & x(s) - x(s-1) \end{pmatrix} \tag{4.7}$$

- Fonction mode($\mathbf{x}$) : cette opération donne la valeur qui a l'occurrence la plus élevée dans le vecteur $\mathbf{x}$.

Exemple 4.32.

Prenons l'exemple du code convolutif $\mathcal{C}(3, 2, 3)$ défini par la matrice génératrice :

$$\mathbf{G} = \begin{pmatrix} 7 & 4 & 1 \\ 2 & 5 & 7 \end{pmatrix}$$

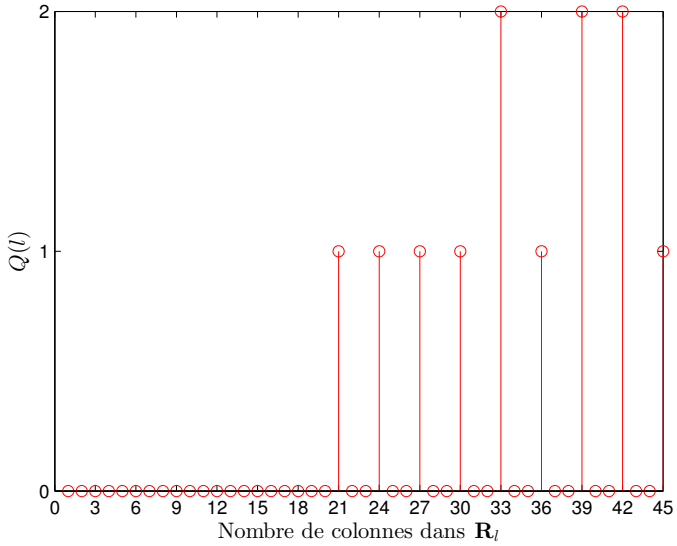


Figure 4.1 — Nombre de colonnes presque dépendantes dans $\tilde{\mathbf{R}}_l$ codée par $\mathcal{C}(3,2,3)$ pour $p_e = 0.01$

Sur la figure 4.1, nous représentons le nombre de colonnes presque dépendantes $Q(l)$ pour différentes valeurs de l et une probabilité d'erreur du canal $p_e = 0.01$. On peut remarquer que peu de colonnes presque dépendantes ont été détectées. Nous indiquons dans le tableau ci-dessous les ensembles $\mathcal{I}$ et diff($\mathcal{I}$) qui vont être utilisés pour identifier la taille n.

$\mathcal{I}$	21	24	27	30	33	36	39	42	45
diff($\mathcal{I}$)		3	3	3	3	3	3	3	3

Tableau 4.1 — Nombre de colonnes presque dépendantes pour $\mathcal{C}(3,2,3)$ et $p_e = 0.01$

Afin de déterminer la valeur identifiée de n, l'équation (4.5) est utilisée. Cette valeur est égale à 3 qui correspond à la bonne taille des mots de code.

La méthode présentée dans cette section peut être appliquée uniquement aux codes binaires. Par la suite, notre objectif est de présenter des méthodes plus générales qui peuvent être appliquées aux codes binaires et non-binaires. Nous nous intéressons en particulier aux codes en bloc. Mais, nos algorithmes peuvent être également appliqués aux codes convolutifs.

4.2.2 Codes non-binaires

Dans cette section, nous démontrons qu'il est possible de généraliser la technique d'identification aveugle proposée dans [SH05,SHB09] et décrite précédemment au cas des codes en bloc non-binaires à condition que les paramètres du corps de Galois soient connus par le récepteur. Pour atteindre cet objectif, il est nécessaire de détecter les colonnes presque dépendantes dans les matrices composées de symboles du corps GF(2^m) entachés d'erreurs en étudiant la probabilité de détection de ces colonnes, notée P_i. Le calcul de P_i est essentiel afin de déterminer le seuil optimal de détection. Le comportement de ce seuil en fonction de P_i et du nombre de lignes M sera étudié. Nous montrons que cette technique généralisée et la technique mise en oeuvre pour identifier la taille des codes binaires nécessitent la connaissance de la probabilité d'erreur du canal p_e. Pour cette raison, nous proposons deux nouvelles approches plus robustes, car elles nous permettent l'identification aveugle de la taille des mots de code pour des codes en bloc non-binaires et binaires sans utiliser la probabilité d'erreur

p_e. Ces deux nouvelles approches sont basées sur l'analyse des comportements de la moyenne arithmétique et de la variance du nombre de zéros dans les colonnes des matrices construites par l'algorithme d'élimination de Gauss dans GF (2^m).

Afin d'évaluer nos algorithmes d'identification aveugles, nous supposons que les séquences codées sont transmises à travers un canal q-aire symétrique (QSC) qui est le canal le plus simple. Mais, nos algorithmes peuvent fonctionner avec tous les types de canaux. Cependant, il est nécessaire que la probabilité d'erreur symbole p_e calculée à la sortie du démodulateur soit connue dans le cas notre premier algorithme. En effet, les blocs de la chaîne de transmission : le modulateur, le canal de transmission et le démodulateur peuvent être modélisés par un canal non-binaire. Comme mentionné dans le chapitre 2, il existe différents modèles de canaux physiques de transmission. Lorsque les données modulées par une modulation QAM d'ordre q sont transmises sur un canal BBAG, la probabilité d'erreur p_e calculée à la sortie du démodulateur peut être déterminée par l'équation (2.10). Dans [Cra91], une méthode simple pour calculer l'expression théorique de la probabilité d'erreur p_e dans le cas d'une modulation par déplacement de phase (PSK : Phase-shift keying) sur un canal BBAG a été proposée. L'expression théorique de la moyenne de cette probabilité pour un canal de Rayleigh à trajets multiples a été étudiée dans [CSA95] dans le cas d'une modulation PSK d'ordre élevé et dans [LTC98] dans le cas d'une modulation QAM. Dans le contexte de la radio cognitive, le modèle du canal à trajets multiples est le plus utilisé. Ce modèle réaliste engendre des erreurs groupées ou bursts qui peuvent être corrigées par l'utilisation d'un entrelaceur en association avec des codes correcteurs d'erreurs. Dans cette situation, les erreurs à la sortie du désentrelaceur en réception peuvent être modélisées par un canal QSC lorsqu'une opération de décodage à décision ferme est utilisée.

4.2.2.1 Description de la méthode

Le principe de la méthode basée sur le critère du rang illustré dans le chapitre précédent est de chercher le nombre maximum de colonnes linéairement indépendantes dans les matrices $\mathbf{R}_l$ qui sont construites à partir des symboles non-entachés d'erreurs c_i. Cela nous permet de calculer le nombre de colonnes linéairement dépendantes dans la matrice $\mathbf{R}_l$. Dans le cas d'une transmission bruitée, toutes les matrices $\tilde{\mathbf{R}}_l$ construites de la même façon que $\mathbf{R}_l$ en utilisant des symboles reçus entachés d'erreurs r_i peuvent être de rang plein. Une matrice $\tilde{\mathbf{R}}_l$ peut s'exprimer en fonction de $\mathbf{R}_l$ par :

$$\tilde{\mathbf{R}}_l = \mathbf{R}_l + \mathbf{E}_l \tag{4.8}$$

où $\mathbf{E}_l$ est une matrice de taille $(M \times l)$ construite de la même façon que $\mathbf{R}_l$ en utilisant les erreurs introduites par le canal de transmission. Donc, la dépendance des colonnes est perturbée par la présence des erreurs dans quelques symboles reçus. Dans un tel contexte, les auteurs dans [SH05, SHB09] ont proposé, dans le cas des codes binaires, de chercher le nombre de colonnes presque dépendantes dans les matrices composées de bits reçus en utilisant l'algorithme d'élimination de Gauss dans GF(2). Nous avons illustré en détail le principe de cette méthode dans la section 4.2.1. Inspiré par cette idée, il suffit, dans le cas des codes correcteurs d'erreurs non-binaires, d'appliquer l'algorithme d'élimination de Gauss dans GF(2^m) à la matrice $\tilde{\mathbf{R}}_l$ pour obtenir une nouvelle matrice $\tilde{\mathbf{T}}_l$ de taille $(M \times l)$ qui n'est pas forcement une matrice triangulaire. Les sorties de l'algorithme de Gauss sont la matrice $\tilde{\mathbf{T}}_l$ et la matrice $\mathbf{A}_l$, où $\mathbf{A}_l$ décrit les opérations de combinaisons et de permutations de colonnes. Comme dans le cas binaire, cet algorithme peut être défini par l'application linéaire décrite dans l'équation (4.1).

Dans le cas d'une transmission non-bruitée, le nombre de colonnes dépendantes dans $\mathbf{R}_l$, pour $l = \alpha \cdot n$, $\forall \alpha \in \mathbb{N}$, correspond au nombre de colonnes nulles dans la matrice $\mathbf{T}_l$ ($\mathbf{R}_l \cdot \mathbf{A}_l = \mathbf{T}_l$). La forme de la matrice $\mathbf{T}_l$ est décrite sur la figure 4.2.

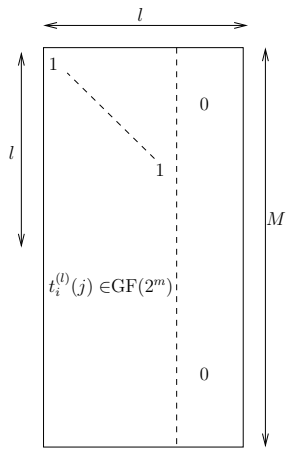


Figure 4.2 — Forme de la matrice $\mathbf{T}_l$ pour $l = \alpha \cdot n$

En raison de la présence des erreurs introduites par le canal dans $\tilde{\mathbf{R}}_l$, pour $l = \alpha \cdot n$, où $\alpha \in \mathbb{N}$, les colonnes de $\tilde{\mathbf{T}}_l$ qui correspondent aux colonnes presque dépendantes dans $\tilde{\mathbf{R}}_l$ contiennent certains symboles non-nuls. En utilisant les équations (4.1) et (4.8), la matrice $\tilde{\mathbf{T}}_l$ peut s'écrire sous la forme :

$$\tilde{\mathbf{T}}_l = \mathbf{T}_l + \mathbf{E}_l \cdot \mathbf{A}_l \tag{4.9}$$

L'idée de nos algorithmes est d'étudier le nombre de zéros dans les colonnes de $\tilde{\mathbf{T}}_l$ afin de détecter les colonnes presque dépendantes dans $\tilde{\mathbf{R}}_l$.

Notons $B_l(i)$ le nombre de zéros dans la colonne $\tilde{\mathbf{t}}_i^{(l)}$. Cette variable sera étudiée en fonction de l en supposant que les bits représentant un élément du corps $\mathrm{GF}(q)$, pour $q = 2^m$, sont indépendants et uniformément distribués.

- Pour $l \neq \alpha \cdot n$: les matrices $\tilde{\mathbf{R}}_l$ se comportent comme des matrices non-binaires aléatoires. Alors, elles sont souvent de rang plein et il n'y a pas de combinaison linéaire entre les colonnes. Dans ce cas, $B_l(i)$, pour tout $i \in [\![1, l]\!]$, suit une loi binomiale $\mathcal{B}(M, 1/q)$ de moyenne égale à M/q.

- Pour $l = \alpha \cdot n$: il existe normalement au moins une combinaison linéaire dans le cas d'une transmission non-bruitée. La variable $B_l(i)$ sera égale à M pour i une position de colonne dépendante. Dans le cas d'une transmission entachée d'erreurs, cette variable aura deux comportements possibles en fonction de i.

  - Si i n'est pas une position de colonne presque dépendante : la variable $B_l(i)$ suivra une loi binomiale $\mathcal{B}(M, 1/q)$ de paramètres M et $1/q$ comme dans le cas de $l \neq \alpha \cdot n$.

  - Si i est une position de colonne presque dépendante : la variable $B_l(i)$ suivra une loi binomiale de paramètres M et P_i, notée $\mathcal{B}(M, P_i)$. Le paramètre P_i correspond à la probabilité d'avoir un coefficient $\tilde{t}_i^{(l)}(j)$ égal à 0 (i.e. $P_i = Pr\left[\tilde{t}_i^{(l)}(j) = 0\right]$).

Dans le cas des codes binaires, la probabilité P_i a été calculée dans [Val00] et elle est définie par l'équation (4.2). Cependant, elle n'a jamais été étudiée dans le cas général de codes dans GF (q). Le calcul de cette probabilité est indispensable afin de détecter les colonnes presque dépendantes dans $\tilde{\mathbf{R}}_l$ en délimitant les deux comportements de $B_l(i)$ en fonction de i. Notre objectif est d'étudier la probabilité P_i dans le cas des codes non-binaires. Par la suite, nous présentons l'étude théorique de P_i.

4.2.2.2 Calcul de la probabilité P_i d'avoir 0 dans la colonne $\mathbf{t}_l^{(i)}$

Pour $l = n$ et i une position de colonne presque dépendante, le coefficient $\tilde{t}_i^{(l)}(j)$ de la colonne $\tilde{\mathbf{t}}_i^{(l)}$ peut être obtenu, en utilisant l'équation (4.9), par :

$$\tilde{t}_i^{(l)}(j) = t_i^{(l)}(j) + \sum_{k=1}^{n} a_i^{(l)}(k) \cdot e_k^{(l)}(j) = \sum_{k=1}^{n} a_i^{(l)}(k) \cdot e_k^{(l)}(j) \tag{4.10}$$

où $t_i^{(l)}(j) = 0$. Dans le cas d'une transmission non-bruitée, la somme $\sum_{k=1}^{n} a_i^{(l)}(k) \cdot e_k^{(l)}(j)$ est nulle car $e_k^{(l)}(j) = 0$, $\forall k \in \{1, \cdots, n\}$ et $\forall j \in \{1, \cdots, M\}$. En revanche, dans le cas d'une transmission bruitée, les coefficients $e_i^{(l)}(j) \in \mathrm{GF}(2^m)$ correspondent aux erreurs introduites par le canal et additionnées aux symboles $r_i^{(l)}(j) \in \mathrm{GF}(2^m)$ afin de générer des symboles bruités $\tilde{r}_i^{(l)}(j) \in \mathrm{GF}(2^m)$. Notre objectif est de déterminer P_i la probabilité de détecter un coefficient nul dans la colonne $\tilde{\mathbf{t}}_i^{(l)}$. Cette probabilité peut être définie par la probabilité d'avoir $\sum_{k=1}^{n} a_i^{(l)}(k) \cdot e_k^{(l)}(j) = 0$ (i.e. $P_i = Pr\left[\sum_{k=1}^{n} a_i^{(l)}(k) \cdot e_k^{(l)}(j) = 0\right]$). Notons $N_i(l)$ le nombre minimal de combinaisons linéaires requises pour obtenir la colonne $\tilde{\mathbf{t}}_i^{(l)}$. Alors, il peut exister des positions parmi $N_i(l)$ où $e_i^{(l)}(j) = 0$. Ainsi, P_i peut être définie par la probabilité d'avoir $\sum_{k=1}^{s} a_i^{(l)}(k) \cdot e_k^{(l)}(j) = 0$ où s correspond au nombre de positions parmi $N_i(l)$ où $e_i^{(l)}(j) \neq 0$:

$$P_i = Pr\left[X = 0\right] + \sum_{s=1}^{N_i(l)} Pr\left[X = s, \sum_{k=1}^{s} a_i^{(l)}(k) \cdot e_k^{(l)}(j) = 0\right] \tag{4.11}$$

où X est une variable aléatoire qui représente le nombre de positions parmi $N_i(l)$ où $e_i^{(l)}(j) \neq 0$. En effet, nous montrons dans l'annexe E que la probabilité P_i d'avoir $\tilde{t}_i^{(l)}(j) = 0$ peut être déterminée par :

$$P_i = \frac{1 + (q-1) \cdot \left(1 - \dfrac{p_e \cdot q}{q-1}\right)^{N_i(l)}}{q} \tag{4.12}$$

Dans le cas de GF(2) (i.e. $q = 2$), cette probabilité peut s'écrire sous la forme :

$$P_i = \frac{1 + (1 - 2 \cdot p_e)^{N_i(l)}}{2} \tag{4.13}$$

On peut vérifier que cette expression correspond bien à l'équation (4.2).

Après avoir étudié les comportements de la variable $B_l(i)$ et le calcul de la probabilité P_i, nous décrivons par la suite le principe des trois méthodes proposées pour identifier en aveugle la taille des mots de code pour des codes dans $\mathrm{GF}(2^m)$.

4.2.3 Les algorithmes d'identification

Dans cette partie, nous présentons trois méthodes permettant d'identifier la taille des mots de code d'un code linéaire non-binaire dans un environnement bruité. Les trois méthodes sont basées sur le principe de la recherche des matrices $\tilde{\mathbf{R}}_l$ de rang déficient, qui correspondent aux matrices ayant au moins une colonne presque dépendante. Dans la première méthode, l'idée est de délimiter les comportements de $B_l(i)$ en déterminant un seuil optimal qui dépend de p_e. Dans la deuxième et la troisième méthode, des calculs statistiques sont utilisés pour détecter des matrices de rang déficient sans connaître la probabilité d'erreur p_e. Pour la deuxième méthode le critère de la moyenne arithmétique est appliqué à $B_l(i)$ et pour la troisième méthode, le critère de la variance

est utilisé.

4.2.3.1 Algorithme basé sur le nombre de colonnes presque dépendantes

L'identification de la dimension d'un espace vectoriel engendré par un code $\mathcal{C}$ est équivalente à trouver celle d'un espace vectoriel engendré par son code dual $\mathcal{C}^{\perp}$. Pour tout vecteur $\mathbf{h}$ appartenant à $\mathcal{C}^{\perp}$ et pour tout mot de code $\mathbf{r}$ de $\mathcal{C}$, la relation entre les deux est définie par l'expression $\mathbf{r}\cdot\mathbf{h}^T = 0$. Dans des conditions de transmission non-bruitée, la matrice $\mathbf{R}_n$, pour $l = n$, qui est composée de M mots de code de longueur n, doit satisfaire :

$$\mathbf{R}_n \cdot \mathbf{h}^T = \mathbf{0} \tag{4.14}$$

On peut remarquer que $\mathbf{h}$ appartient au noyau de $\mathbf{R}_n$, noté $\ker(\mathbf{R}_n)$. Alors, on a $\mathcal{C}^{\perp} \subset \ker(\mathbf{R}_n)$. Puisqu'une colonne nulle $\mathbf{t}_i^{(n)}$ dans la matrice $\mathbf{T}_n$ est obtenue par la multiplication de la matrice $\mathbf{R}_n$ composée des mots de code avec une colonne $\mathbf{a}_i^{(n)}$, cette dernière vérifie alors la relation définie par l'équation (4.14). De ce fait, la colonne $\mathbf{a}_i^{(n)}$ appartient à l'espace du code dual $\mathcal{C}^{\perp}$. Donc, trouver des colonnes dépendantes dans $\mathbf{R}_l$ est équivalent à trouver des colonnes $\mathbf{a}_i^{(l)}$ qui appartiennent au code dual $\mathcal{C}^{\perp}$. Dans une transmission bruitée, un vecteur $\mathbf{h}$ représente une équation de parité (i.e. $\mathbf{h} \in \mathcal{C}^{\perp}$) avec une grande probabilité si la relation $\tilde{\mathbf{R}}_l \cdot \mathbf{h}^T$ a un faible poids de Hamming [Bar07]. On peut conclure que $\mathbf{a}_i^{(l)}$ appartient à $\mathcal{C}^{\perp}$ si la colonne correspondante $\tilde{\mathbf{t}}_i^{(l)} = \tilde{\mathbf{R}}_l \cdot \mathbf{a}_i^{(l)}$ a un faible poids de Hamming, c'est-à-dire un nombre faible d'éléments non-nuls. De ce fait, notre objective est de déterminer les colonnes $\tilde{\mathbf{t}}_i^{(l)}$ qui ont un nombre élevé de zéros.

La variable $B_l(i)$ a deux comportements qui dépendent de l'appartenance ou non de la colonne $\mathbf{a}_i^{(l)}$ au code dual $\mathcal{C}^{\perp}$:

- si le vecteur $\mathbf{a}_i^{(l)}$ n'appartient pas au code dual : $B_l(i)$ suit une loi binomiale $\mathcal{B}(M, 1/q)$ de moyenne M/q. Le nombre moyen de zéros contenus dans la i-ème colonne sera proche de $\dfrac{M}{q}$.
- si le vecteur $\mathbf{a}_i^{(l)}$ appartient au code dual : $B_l(i)$ suit une loi binomiale $\mathcal{B}(M, P)$ de moyenne $M \cdot P_i$. Le nombre moyen de zéros contenus dans la i-ème colonne sera proche de $M \cdot P_i$.

Le nombre minimal de combinaisons linéaires nécessaires pour obtenir la i-ème colonne de la matrice $\tilde{\mathbf{T}}_l$, $N_i(l)$, correspond au poids de Hamming du vecteur $\tilde{\mathbf{t}}_i^{(l)}$. Puisque $\tilde{\mathbf{t}}_i^{(l)}$ et $\mathbf{a}_i^{(l)}$ ont le même poids, alors :

$$N_i(l) = w\left(\mathbf{a}_i^{(l)}\right) \tag{4.15}$$

Pour délimiter les deux comportements de la variable $B_l(i)$, il est judicieux de fixer un seuil, noté η, et décider si $\mathbf{a}_i^{(l)}$ appartient ou non au code dual. Dans ce cas, nous définissons deux hypothèses $\mathcal{H}_0$ et $\mathcal{H}_1$ tel que :

$$\mathcal{H}_0 \text{ si } \mathbf{a}_i^{(l)} \in \mathcal{C}^{\perp} \text{ et } \mathcal{H}_1 \text{ si } \mathbf{a}_i^{(l)} \notin \mathcal{C}^{\perp} \tag{4.16}$$

Les deux comportements de $B_l(i)$ sont délimités comme suit :

$$\begin{cases} \text{Si } B_l(i) > \frac{M}{q} \cdot \eta \text{ alors } \mathcal{H}_0 \\ \text{Si } B_l(i) \leq \frac{M}{q} \cdot \eta \text{ alors } \mathcal{H}_1 \end{cases} \tag{4.17}$$

Maintenant, notre objectif est de calculer un seuil optimal η_{opt} qui permet de minimiser la probabilité de mauvaise détection d'une colonne presque dépendante, notée P_{wd}, qui correspond à la somme de la probabilité de fausse alarme, notée P_{fa}, et de la probabilité de non-détection d'une

colonne théoriquement dépendante, notée P_{nd}. Afin de déterminer les probabilités P_{fa} et P_{nd}, nous allons étudier les comportements de la variable $B_l(i)$ en fonction des deux hypothèses $\mathcal{H}_0$ et $\mathcal{H}_1$:

Sous l'hypothèse $\mathcal{H}_0$: la variable $B_l(i)$ suit une loi binomiale $\mathcal{B}(M, P_i)$. Alors, la probabilité d'avoir la variable $B_l(i)$ supérieure à $\frac{M}{q} \cdot \eta$ est :

$$Pr\left[B_l(i) > \frac{M}{q} \cdot \eta \mid \mathcal{H}_0\right] = \sum_{j=\lfloor \frac{M}{q} \cdot \eta \rfloor + 1}^{M} \binom{M}{j} \cdot P_i^j \cdot (1 - P_i)^{M-j} \tag{4.18}$$

Sous l'hypothèse $\mathcal{H}_1$: la variable $B_l(i)$ suit une loi binomiale $\mathcal{B}(M, 1/q)$. Alors, la probabilité d'avoir la variable $B_l(i)$ inférieure ou égale $\frac{M}{q} \cdot \eta$ est :

$$Pr\left[B_l(i) \leq \frac{M}{q} \cdot \eta \mid \mathcal{H}_1\right] = \sum_{j=0}^{\lfloor \frac{M}{q} \cdot \eta \rfloor} \binom{M}{j} \cdot \frac{(q-1)^{M-j}}{q^M} \tag{4.19}$$

En utilisant ces deux probabilités, nous allons calculer la probabilité de fausse alarme P_{fa}, la probabilité de non-détection P_{nd} et la probabilité de détection, notée P_{det}.

Calcul de la probabilité de fausse alarme P_{fa} : cette probabilité correspond à décider que la colonne $\mathbf{a}_i^{(l)}$ appartient au code dual $\mathcal{C}^{\perp}$ alors qu'en réalité elle n'y appartient pas. Cette probabilité peut être déterminée par :

$$P_{fa} = Pr\left[B_l(i) > \frac{M}{q} \cdot \eta \mid \mathcal{H}_1\right] = \sum_{j=\lfloor \frac{M}{q} \cdot \eta \rfloor + 1}^{M} \binom{M}{j} \cdot \frac{(q-1)^{M-j}}{q^M} \tag{4.20}$$

Calcul de la probabilité de non-détection P_{nd} : cette probabilité correspond à décider que la colonne $\mathbf{a}_i^{(l)}$ n'appartient pas à $\mathcal{C}^{\perp}$ alors que elle y appartient. Cette probabilité correspond à :

$$P_{nd} = Pr\left[B_l(i) \leq \frac{M}{q} \cdot \eta \mid \mathcal{H}_0\right] = \sum_{j=0}^{\lfloor \frac{M}{q} \cdot \eta \rfloor} \binom{M}{j} \cdot P_i^j \cdot (1 - P_i)^{M-j} \tag{4.21}$$

Calcul de la probabilité de détection P_{det} : cette probabilité est définie par :

$$P_{det} = 1 - P_{nd} = Pr\left[B_l(i) > \frac{M}{q} \cdot \eta \mid \mathcal{H}_0\right] = \sum_{j=\lfloor \frac{M}{q} \cdot \eta \rfloor + 1}^{M} \binom{M}{j} \cdot P_i^j \cdot (1 - P_i)^{M-j} \tag{4.22}$$

En utilisant les équations (4.20) et (4.22), le seuil optimal peut être déterminé par :

$$\begin{aligned} \eta_{opt} &= \arg\min_{\eta}(P_{wd}) = \arg\min_{\eta}(P_{nd} + P_{fa}) = \arg\min_{\eta}(1 + P_{fa} - P_{det}) \\ &= \arg\min_{\eta}\left(1 + \sum_{j=\left\lfloor \frac{M}{q} \cdot \eta \right\rfloor + 1}^{M} \binom{M}{j} \cdot \left[q^{-M} \cdot (q-1)^{M-j} - P_i^j \cdot (1 - P_i)^{M-j}\right]\right) \end{aligned} \tag{4.23}$$

Nous pouvons remarquer que le seuil optimal η_{opt} dépend des paramètres : M, q et P_i. Or, d'après l'équation (4.12), le paramètre P_i dépend de la probabilité d'erreur p_e et du poids de Hamming $w\left(\mathbf{a}_i^{(l)}\right)$. Donc, nous allons étudier la probabilité de mauvaise détection d'une colonne presque dépendante en fonction du seuil.

La probabilité de mauvaise détection P_{wd} en fonction du seuil η est représenté sur la figure 6.3 pour les paramètres suivants : $q = 2^4$, $M = 500$, $p_e = 0.01$ et $w\left(\mathbf{a}_i^{(l)}\right) = 20$.

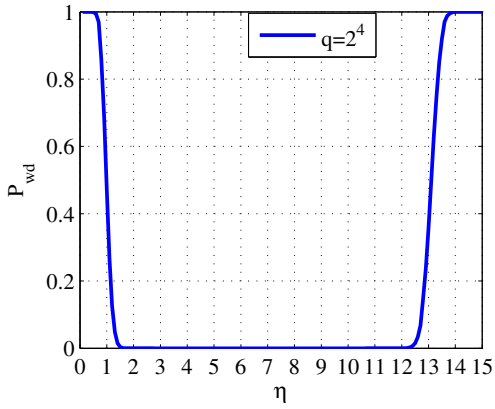


Figure 4.3 — Probabilité P_{wd} en fonction du seuil η

Nous observons que la probabilité P_{wd} est quasiment nulle lorsque le seuil $\eta \in [\eta_{min}, \eta_{max}] = [2, 12]$, où η_{min} et η_{max} sont définis par les bornes minimale et maximale de l'intervalle dans lequel la probabilité P_{wd} est minimale. En revanche, notre objectif est de chercher une valeur précise du seuil optimal qui correspond à la valeur minimale de P_{wd} et permet de délimiter les comportements de la variable $B_l(i)$. L'idée est donc de calculer la dérivée première de P_{wd} et de trouver la solution de l'équation :

$$\frac{\partial P_{wd}(\eta_{opt})}{\partial \eta} = 0 \tag{4.24}$$

Afin de simplifier le calcul, une loi normale peut être utilisée pour approcher les lois binomiales de $B_l(i)$ lorsque M est grand [Bog05]. Ainsi, les deux comportements suivront :

- Si $\mathbf{a}_i^{(l)} \in \mathcal{C}^{\perp}$:

$$B_l(i) \to \mathcal{N}\left(\mu_0, \sigma_0^2\right) \tag{4.25}$$

- Si $\mathbf{a}_i^{(l)} \notin \mathcal{C}^{\perp}$:

$$B_l(i) \to \mathcal{N}\left(\mu_1, \sigma_1^2\right) \tag{4.26}$$

où $\mathcal{N}\left(\mu_0, \sigma_0^2\right)$ est une loi normale de paramètres $\mu_0 = M \cdot P_i$ et $\sigma_0^2 = M \cdot P_i \cdot (1 - P_i)$ et $\mathcal{N}\left(\mu_1, \sigma_1^2\right)$ correspond à une loi normale de paramètres $\mu_1 = M/q$ et $\sigma_1^2 = M \cdot (q-1)/q^2$. Dans cette configuration, le test statistique suivant peut être utilisé pour distinguer les deux comportements :

$$\begin{cases} \text{Si } B_l(i) > \hat{\eta} \text{ alors } \mathcal{H}_0 \\ \text{Si } B_l(i) \leq \hat{\eta} \text{ alors } \mathcal{H}_1 \end{cases} \tag{4.27}$$

en posant $\hat{\eta} = \frac{M}{q} \cdot \eta$ un nombre réel de l'intervalle $[0, M]$.

Alors, la valeur optimale du seuil $\hat{\eta}$ minimisant la probabilité de mauvaise détection P_{wd} peut être calculée par :

$$\hat{\eta}_{opt} = \arg\min_{\hat{\eta}} \left(1 - \phi\left(\frac{\hat{\eta} - \mu_1}{\sigma_1}\right) + \phi\left(\frac{\hat{\eta} - \mu_0}{\sigma_0}\right)\right) \tag{4.28}$$

où $\phi(x)$ est la fonction de répartition de la loi normale réduite :

$$\phi(x) = \frac{1}{\sqrt{2 \cdot \pi}} \cdot \int_{-\infty}^{x} e^{-\frac{t^2}{2}} \cdot dt \tag{4.29}$$

Afin de trouver une valeur précise de $\hat{\eta}_{opt}$, la dérivée première de P_{wd} est calculée. Le lecteur intéressé pourra se référer à l'annexe F où un calcul détaillé de la dérivée est présentée. Cette dérivée sera nulle lorsque $\hat{\eta}_{opt}$ est une solution de l'équation du second degré ci-dessous :

$$a \cdot \hat{\eta}_{opt}^2 + b \cdot \hat{\eta}_{opt} + c = 0 \tag{4.30}$$

avec :

$$\begin{cases} a = \sigma_1^2 - \sigma_0^2 \\ b = 2 \cdot \left(\mu_1 \cdot \sigma_0^2 - \mu_0 \cdot \sigma_1^2\right) \\ c = \mu_0^2 \cdot \sigma_1^2 - \mu_1^2 \cdot \sigma_0^2 - 2 \cdot \sigma_0 \cdot \sigma_1 \cdot \ln\left(\dfrac{\sigma_1}{\sigma_0}\right) \end{cases}$$

En remplaçant les paramètres μ_0, μ_1, σ_0^2 et σ_1^2 par leurs valeurs, les coefficients a, b et c peuvent s'écrire :

$$\begin{cases} a = \dfrac{M}{q^2} \cdot \left(q - 1 - q^2 \cdot P_i \cdot (1 - P_i)\right) \\ b = \dfrac{2 \cdot M^2}{q^2} \cdot (1 - P_i^2) \\ c = \dfrac{M^3}{q^3} \cdot P_i \cdot (q \cdot P_i - 1) - \dfrac{M}{q} \cdot \sqrt{P_i \cdot (1 - P_i) \cdot (q - 1)} \cdot \ln\left(\dfrac{q - 1}{q^2 \cdot (1 - P_i) \cdot P_i}\right) \end{cases}$$

Impact des paramètres utilisés sur le seuil de décision

Dans le cadre de cette étude, la variation du paramètre M n'a aucun effet en pratique sur la valeur du seuil $\hat{\eta}$. Pour évaluer l'impact du paramètre p_e sur la valeur du seuil, nous représentons sur la figure 4.4 la probabilité de mauvaise détection P_{wd} en fonction de $\hat{\eta}/M$ et de p_e en supposant $q = 2^3$ et $w\left(\mathbf{a}_i^{(l)}\right) = 20$. Pour chaque valeur de p_e, le seuil optimal $\hat{\eta}_{opt}$ qui correspond à la racine de l'équation (4.30) est calculé. Les valeurs normalisées calculées de $\hat{\eta}_{opt}/M$ et de l'intervalle $[\hat{\eta}_{min}/M, \hat{\eta}_{max}/M]$, dans lequel P_{wd} est proche de zéro, sont indiquées dans le tableau 4.2.

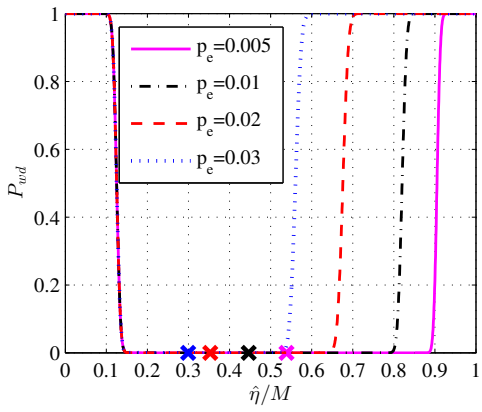


Figure 4.4 — $P_{wd} = f(\hat{\eta}/M, p_e)$ pour $q = 2^3$, $M = 2000$ et $w\left(\mathbf{a}_i^{(l)}\right) = 20$

p_e	$\hat{\eta}_{opt}/M$	$[\hat{\eta}_{min}/M, \hat{\eta}_{max}/M]$
0.005	0.5388	$[0.1865, 0.8795]$
0.01	0.4468	$[0.1865, 0.788]$
0.02	0.3532	$[0.1865, 0.637]$
0.03	0.2991	$[0.1865, 0.519]$

Tableau 4.2 — Seuil optimal $\hat{\eta}_{opt}$ et intervalle du seuil en fonction de p_e

D'après la figure 4.4, nous pouvons déduire que l'intervalle du seuil qui satisfait $P_{wd} \approx 0$ diminue lorsque la valeur de p_e augmente. Pour toute probabilité d'erreur p_e, le seuil minimal $\hat{\eta}_{min}$ est stable. Par contre, le seuil maximal $\hat{\eta}_{max}$ dépend de la valeur p_e. En effet, en utilisant le tableau 4.2, nous pouvons vérifier que le seuil optimal $\hat{\eta}_{opt}$ et $\hat{\eta}_{max}$ diminuent lorsque la probabilité d'erreur p_e augmente.

Pour $p_e = 0.01$, nous montrons sur la figure 4.5 l'évolution de la probabilité de mauvaise détection en fonction du seuil pour différentes valeurs de $w\left(\mathbf{a}_i^{(l)}\right)$. Dans le tableau 4.3, nous présentons l'intervalle $[\hat{\eta}_{min}/M, \hat{\eta}_{max}/M]$ et les valeurs de $\hat{\eta}_{opt}/M$ en fonction de $w\left(\mathbf{a}_i^{(l)}\right)$.

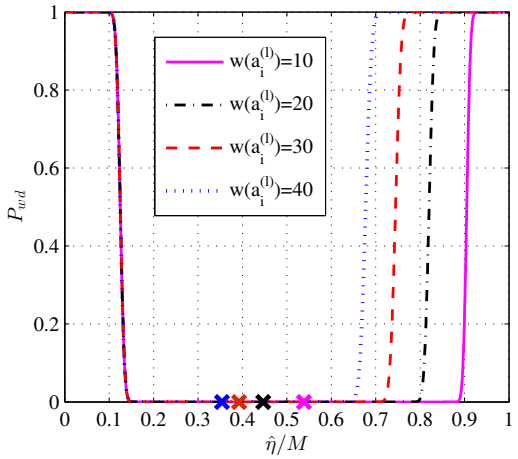


Figure 4.5 — $P_{wd} = f\left(\hat{\eta}/M, w\left(\mathbf{a}_i^{(l)}\right)\right)$ pour $M = 2000$, $p_e = 0.01$ et $q = 2^3$

$w\left(\mathbf{a}_i^{(l)}\right)$	$\hat{\eta}_{opt}/M$	$[\hat{\eta}_{min}/M, \hat{\eta}_{max}/M]$
10	0.5384	$[0.1535, 0.879]$
20	0.4468	$[0.1535, 0.788]$
30	0.3924	$[0.1535, 0.708]$
40	0.3539	$[0.1535, 0.6385]$

Tableau 4.3 — Seuil optimal $\hat{\eta}_{opt}$ et intervalle du seuil en fonction de $w\left(\mathbf{a}_i^{(l)}\right)$

D'après la figure 4.5 et le tableau 4.3, nous pouvons observer une influence significative du paramètre $w\left(\mathbf{a}_i^{(l)}\right)$ sur la valeur optimale du seuil $\hat{\eta}_{opt}/M$ et l'intervalle $[\hat{\eta}_{min}/M, \hat{\eta}_{max}/M]$. Lorsque la valeur de $w\left(\mathbf{a}_i^{(l)}\right)$ augmente, le seuil $\hat{\eta}_{opt}$ et le seuil maximal $\hat{\eta}_{max}$ diminuent. Cependant, le seuil minimal $\hat{\eta}_{min}$ n'est pas influencé par $w\left(\mathbf{a}_i^{(l)}\right)$.

Sur la figure 4.6, la probabilité P_{wd} en fonction du seuil $\hat{\eta}/M$ est représentée pour $q \in \{4, 8, 16, 32\}$. Nous constatons que l'intervalle du seuil s'agrandit en augmentant le cardinal du corps de Galois q, alors que le seuil optimal $\hat{\eta}_{opt}$ diminue. Pour chaque valeur de q, l'intervalle $[\hat{\eta}_{min}/M, \hat{\eta}_{max}/M]$ et les valeurs de $\hat{\eta}_{opt}/M$ sont récapitulés dans le tableau 4.4. D'après la figure 4.6 et le tableau 4.4, nous constatons que le seuil optimal $\hat{\eta}_{max}$ est constant lorsque q augmente. Cependant, la valeur de $\hat{\eta}_{min}$ est influencée par la variation de ce paramètre.

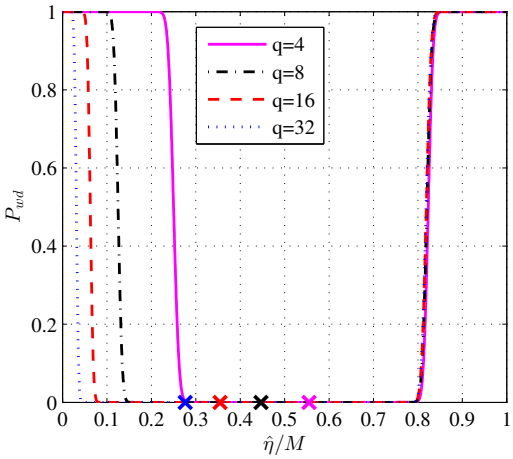


Figure 4.6 — $P_{wd} = f(\hat{\eta}/M, q)$ pour $M = 2000$, $p_e = 0.01$ et $w\left(\mathbf{a}_i^{(l)}\right) = 20$

q	$\hat{\eta}_{opt}/M$	$[\hat{\eta}_{min}/M, \hat{\eta}_{max}/M]$
4	0.5549	$[0.2865, 0.7895]$
8	0.4468	$[0.1565, 0.7895]$
16	0.3545	$[0.085, 0.7895]$
32	0.2761	$[0.0475, 0.7895]$

Tableau 4.4 — Seuil optimal $\hat{\eta}_{opt}$ et intervalle du seuil en fonction de q

Dans cette partie, nous avons étudié, par des exemples, l'impact des paramètres p_e, $w\left(\mathbf{a}_i^{(l)}\right)$ et q sur le seuil optimal normalisé $\hat{\eta}_{opt}/M$ et sur l'intervalle $[\hat{\eta}_{min}/M, \hat{\eta}_{max}/M]$ afin de montrer que ce seuil est sensible à la variation de ces paramètres. De ce fait, la détermination du seuil optimal est essentielle pour identifier la taille des mots de code $\tilde{n}$ à partir des comportements de $B_l(i)$.

Description de l'algorithme

Notons $Q(l)$ le nombre de colonnes presque dépendantes pour chaque matrice $\tilde{\mathbf{R}}_l$ tel que :

$$Q(l) = Card\{i \in [\![0, l]\!], B_l(i) > \hat{\eta}_{opt}\} \tag{4.31}$$

En utilisant cette variable, on peut détecter la présence des colonnes presque dépendantes dans $\tilde{\mathbf{R}}_l$. En effet, si toutes les colonnes sont indépendantes alors $Q(l)$ sera égal à zéro. S'il existe quelques colonnes presque dépendantes, alors $Q(l)$ sera non-nul. En présence d'erreurs de transmission, il est possible de trouver des colonnes presque dépendantes pour des matrices de taille $l \neq \alpha \cdot n$, où $\alpha \in \mathbb{N}$. Dans ce cas, la variable $Q(l)$ sera non-nulle.

Afin d'identifier la taille des mots de code, l'idée est de trouver la distance dont l'occurrence est

la plus élevée dans l'ensemble, noté $\mathcal{I}$, défini par :

$$\mathcal{I} = \{l = 1, \cdots, l_{max} | Q(l) \neq 0\} \tag{4.32}$$

Ainsi, la taille identifiée des mots de code, notée $\tilde{n}$, est obtenue par :

$$\tilde{n} = \text{mode}(\text{diff}(\mathcal{I})) \tag{4.33}$$

Un bref résumé de la méthode basée sur le nombre de colonnes presque dépendantes est présenté dans l'algorithme 2.

Algorithme 2: Algorithme basé sur le nombre de colonnes presque dépendantes

Entrées : $\tilde{\mathbf{r}}$, M, q et p_e
Sorties : La taille identifiée de mots de code $\tilde{n}$

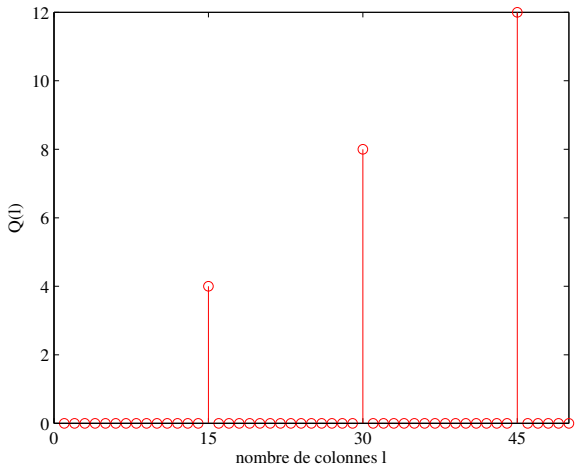
```
pour l = 1 à l_max faire
    Construire la matrice R̃_l de taille (M × l)
    R̃_l → T̃_l = R̃_l · A_l
    pour i = 1 à l faire
        Compter B_l(i)
        Calculer P (4.12)
        Calculer η̂_opt (4.30)
        si B_l(i) > η̂_opt alors
            Q(l) = Q(l) + 1
        fin
    fin
fin
Construire l'ensemble I (4.32)
Déterminer ñ (4.33)
```

Exemple 4.33.

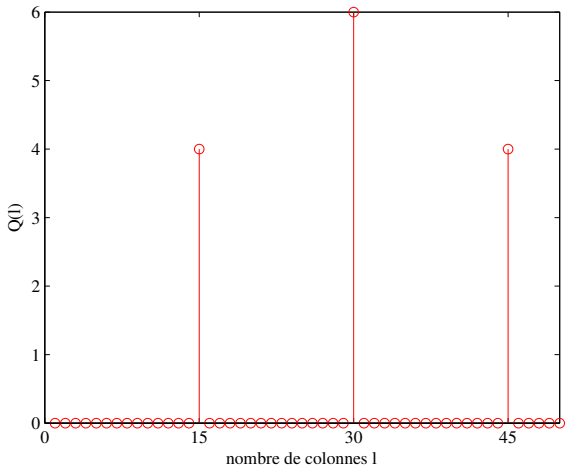
Considérons le code de Reed-Solomon $RS(15, 11)$ dans GF(2^4). Nous essayons d'identifier la taille des mots de code dans le cas d'une probabilité d'erreur $p_e = 0$ et ensuite $p_e = 0.01$. Pour $p_e = 0$, nous représentons sur la figure 4.7(a) le nombre de colonnes presque dépendantes $Q(l)$ en fonction de $l \in [\![1, 50]\!]$. A travers cette figure, nous pouvons vérifier que $Q(l) \neq 0$ pour des matrices $\tilde{\mathbf{R}}_l$ de taille $l = \alpha \cdot n$. En effet, il existe quelques colonnes presque dépendantes dans chaque matrice $\tilde{\mathbf{R}}_{\alpha \cdot n}$. Leur nombre est déterminé par :

$$Q(l) = l - rang\left(\tilde{\mathbf{R}}_l\right) = \alpha \cdot n - rang\left(\tilde{\mathbf{R}}_{\alpha \cdot n}\right) = \alpha \cdot (n - k)$$

Pour $p_e = 0.01$, nous représentons sur la figure 4.7(b) le nombre $Q(l)$ en fonction de l. Pour détecter les colonnes presque dépendantes, il est souhaitable de calculer le seuil optimal donné par l'équation (4.30) pour chaque vecteur $\mathbf{a}_i^{(l)}$ afin de délimiter les deux comportements de la variable $B_l(i)$. Nous pouvons vérifier que $Q(l)$ est non-nul pour les matrices de taille $l = \alpha.n$, mais nous remarquons sur cet exemple que $Q(\alpha \cdot n) < \alpha \cdot (n - k)$.



(a) Cas $p_e = 0$



(b) Cas $p_e = 0.01$

Figure 4.7 — Nombre de colonnes presque dépendantes pour le code $RS(15, 11)$

Nous récapitulons, pour $p_e = 0.01$, dans le tableau 4.5 les tailles des matrices pour lesquelles $Q(l) \neq 0$ (i.e. l'ensemble $\mathcal{I}$ défini dans (4.32)) ainsi que le nombre de colonnes presque dépendantes (i.e. la variable $Q(l)$ définie dans (4.31)). En utilisant l'équation (4.33), la valeur identifiée de n est 15.

$l \in \mathcal{I}$	15	30	45
$Q(l)$	4	6	4
diff($\mathcal{I}$)		15	15

Tableau 4.5 — Nombre de colonnes presque dépendantes pour $p_e = 0.01$

Dans la méthode présentée dans cette partie, la détermination du nombre de colonnes presque dépendantes ($Q(l)$) est basée sur le calcul des $B_l(i)$ et du seuil optimal $\hat{\eta}_{opt}$. Cependant, ce seuil dépend de la valeur de la probabilité d'erreur du canal p_e qui est inconnue pour le récepteur. Donc, il faudrait estimer ce paramètre avant d'appliquer cette méthode d'identification. Une deuxième technique d'identification basée sur le critère de la moyenne arithmétique qui ne nécessite pas

l'estimation de p_e est illustrée dans la prochaine partie.

4.2.3.2 Algorithme basé sur le critère de la moyenne

La moyenne arithmétique des variables $B_l(i)$, où $i \in [\![1, l]\!]$, notée E_l, est définie par :

$$E_l = \frac{\sum_{i=1}^{l} B_l(i)}{l} \tag{4.34}$$

Définition 4.18. *Soient $X_1, X_2, \cdots, X_m$ des variables aléatoires indépendantes qui suivent respectivement les lois :*

$$\mathcal{N}\left(\mu_1, \sigma_1^2\right), \mathcal{N}\left(\mu_2, \sigma_2^2\right), ..., \mathcal{N}\left(\mu_m, \sigma_m^2\right)$$

Alors, la moyenne définie par $\frac{(X_1+X_2+\cdots+X_m)}{m}$ suit la loi :

$$\mathcal{N}\left(\frac{\mu_1 + \mu_2 + \cdots + \mu_m}{m}, \frac{\sigma_1^2 + \sigma_2^2 + \cdots + \sigma_m^2}{m^2}\right) \tag{4.35}$$

Nous rappelons que la variable $B_l(i)$ qui est le nombre de zéros dans la i-ème colonne de la matrice $\tilde{\mathbf{T}}_l$ a deux comportements possibles en fonction de l :

- Si $l \neq \alpha \cdot n$, pour $\alpha \in \mathbb{N}$, la variable $B_l(i)$ suit la loi normale $\mathcal{N}\left(\mu_1, \sigma_1^2\right)$ pour toutes les colonnes i de $\tilde{\mathbf{T}}_l$. En utilisant la définition 4.18, la moyenne E_l suivra :

$$E_l \to \mathcal{N}\left(\mu_1, \frac{\sigma_1^2}{l}\right) \tag{4.36}$$

  Dans ce cas, la moyenne sera proche de M/q.

- Si $l = \alpha \cdot n$, pour $\alpha \in \mathbb{N}$:
  - Si la i-ème colonne est presque dépendante, la variable $B_l(i)$ suit une loi normale de paramètres $\mathcal{N}\left(\mu_0, \sigma_0^2\right)$.
  - Si la i-ème colonne n'est pas presque dépendante, la variable $B_l(i)$ suit une loi normale de paramètres $\mathcal{N}\left(\mu_1, \sigma_1^2\right)$.

  Ainsi, pour $l = \alpha \cdot n$, la moyenne E_l suit la loi :

$$E_l \to \mathcal{N}\left(\frac{Q(l) \cdot \mu_0 + k_l \cdot \mu_1}{l}, \frac{Q(l) \cdot \sigma_0^2 + k_l \cdot \sigma_1^2}{l^2}\right) \tag{4.37}$$

  où $Q(l)$ est le nombre de colonnes presque dépendantes dans la matrice $\tilde{\mathbf{R}}_l$ et $k_l = l - Q(l)$ est le nombre de colonnes indépendantes dans la même matrice. Nous mentionnons que le nombre de colonnes presque dépendantes $Q(l)$ est déterminé par l'équation (4.31). Dans un environnement non-bruité, la moyenne E_l est quasiment stable à :

$$E_l = \frac{M \cdot (q \cdot (n-k) + k)}{q \cdot n} \tag{4.38}$$

Nous remarquons deux comportements de E_l en fonction de $l = \alpha \cdot n$ ou $l \neq \alpha \cdot n$:

$$\begin{cases} \text{Si } l \neq \alpha \cdot n \text{ alors } E_l \approx \frac{M}{q} \\ \text{Si } l = \alpha \cdot n \text{ alors } E_l > \frac{M}{q} \end{cases} \tag{4.39}$$

L'écart entre ces comportements nous permet de trouver les matrices dont le nombre de colonnes l est un multiple de n. Notons $\mathcal{J}$ l'ensemble de valeurs de l lorsque l'écart $E_l - \frac{M}{q} > 0$:

$$\mathcal{J} = \left\{ l = 1, \cdots, l_{max} | E_l - \frac{M}{q} > 0 \right\} \tag{4.40}$$

La taille identifiée de mots de code sera telle que :

$$\tilde{n} = \text{mode}(\text{diff}(\mathcal{J})) \tag{4.41}$$

L'algorithme proposé pour identifier en aveugle la taille des mots de code en se basant sur le calcul de la moyenne arithmétique est résumé dans l'algorithme 3.

Algorithme 3: Algorithme basé sur le critère de la moyenne

Entrées : $\mathbf{r}$, M et q
Sorties : la taille identifiée de mots de code $\tilde{n}$

pour $l = 1$ *à* l_{max} **faire**
  Construire la matrice $\tilde{\mathbf{R}}_l$ de taille $(M \times l)$
  $\tilde{\mathbf{R}}_l \rightarrow \tilde{\mathbf{T}}_l = \tilde{\mathbf{R}}_l \cdot \mathbf{A}_l$
  pour $i = 1$ *à* l **faire**
    Compter $B_l(i)$
  fin
  Calculer E_l
fin
Construire l'ensemble $\mathcal{J}$ (4.40)
Déterminer $\tilde{n}$ (4.41)

Exemple 4.34.

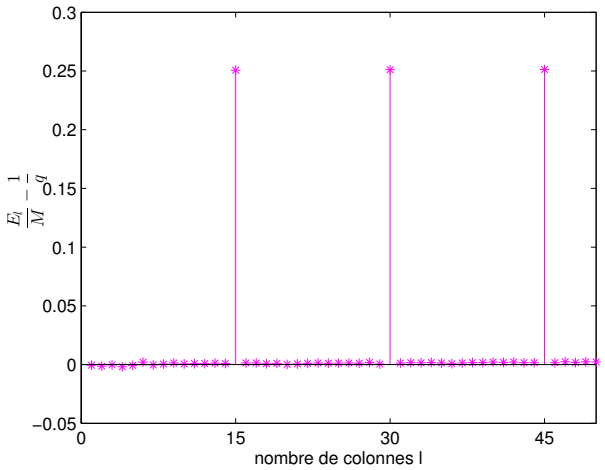
Reprenons l'exemple précédent du code $RS(15, 11)$ dans GF(2^4). La moyenne E_l normalisée par M, où M est fixé à 1000, est représentée sur la figure 4.8. Sur la figure 4.8(a), une probabilité d'erreur nulle (i.e. $p_e = 0$) est considérée. Pour $l \neq \alpha \cdot n$, nous pouvons vérifier que la moyenne E_l normalisée par M est quasiment constante à $1/q = 0.0625$. Pour $l = \alpha \cdot n$, la moyenne E_l satisfait l'équation (4.38) :

$$\frac{1}{q \cdot n} \cdot (q \cdot (n - k) + k) = 0.3125$$

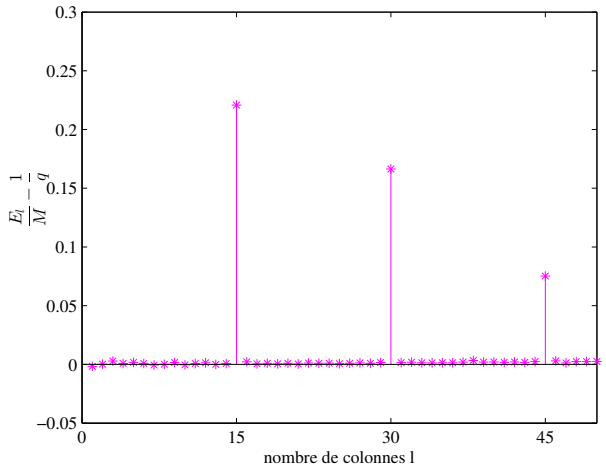
Nous pouvons observer des pics pour $l = \alpha \cdot n$. Ces pics correspondent à une valeur de $\frac{E_l}{M} - \frac{1}{q} = 0.25 > 0$. Sur la figure 4.8(b), l'écart $\frac{E_l}{M} - \frac{1}{q}$ est représenté en fonction de l pour $p_e = 0.01$. En utilisant le définition (4.40), l'ensemble $\mathcal{J}$ est présenté dans le tableau 4.6. Par conséquent, en utilisant l'équation (4.41), la taille identifiée des mots de code est $\tilde{n} = 15$. Nous pouvons constater qu'il existe toujours des pics pour $l = \alpha \cdot n$ dans le cas de $p_e = 0.01$. Les valeurs de ces pics se dégradent par rapport au cas de $p_e = 0$. Cependant, elles permettent une estimation correcte de la taille des mots de code.

$l \in \mathcal{J}$	15	30	45
diff($\mathcal{J}$)		15	15

Tableau 4.6 — Taille de matrices $\tilde{\mathbf{R}}_l$ pour $\frac{E_l}{M} - \frac{1}{q} > 0$



(a) Cas $p_e = 0$



(b) Cas $p_e = 0.01$

Figure 4.8 — Écart entre la moyenne E_l/M et $1/q$

4.2.3.3 Algorithme basé sur le critère de la variance

Dans cette section, nous présentons une méthode d'identification plus robuste et généralisée aux codes correcteurs d'erreurs dans GF(2^m). Cette méthode consiste à calculer la variance du nombre de 0 dans les colonnes des matrices non-binaires $\tilde{\mathbf{T}}_l$ $(B_l(i))$. Notons V_l la variable représentant la variance empirique des $B_l(i)$, où $i \in [\![1, l]\!]$. Cette variable est définie par :

$$V_l = \frac{\sum_{i=1}^{l}(B_l(i) - E_l)^2}{l} \tag{4.42}$$

où E_l est la moyenne arithmétique de $B_l(i)$. La variance V_l a deux comportements différents en fonction de l, $\forall \alpha \in \mathbb{N}$:

- Si $l \neq \alpha \cdot n$ ou $l < n_a$, $B_l(i)$ suit une loi normale de paramètres $\mu_1 = M/q$ et $\sigma_1^2 = M \cdot (q-1)/q^2$,

notée $\mathcal{N}(\mu_1, \sigma_1^2)$, pour toutes les colonnes de $\tilde{\mathbf{T}}_l$. Alors la variance V_l suivra :

$$V_l \rightarrow \frac{\sigma_1^2}{l} \cdot \mathcal{X}_{l-1}^2 \tag{4.43}$$

où $\mathcal{X}_{l-1}^2$ est la loi du chi-2 de paramètre $l-1$. Dans ce cas, la moyenne de V_l normalisée par M^2 sera proche de $(q-1)^2/q^4$.

- Si $l = \alpha \cdot n$ et $l \geq n_a$:
  - Si la i-ème colonne est presque dépendante, $B_l(i)$ suit une loi normale $\mathcal{N}(\mu_0, \sigma_0^2)$ avec $\mu_0 = M \cdot P_i$ et $\sigma_0^2 = M \cdot P_i \cdot (1 - P_i)$.
  - Si la i-ème colonne n'est pas presque dépendante, $B_l(i)$ suit une loi normale $\mathcal{N}(\mu_1, \sigma_1^2)$.

Ainsi, la variable V_l suivra la loi :

$$V_l \rightarrow \frac{\sigma_0^2}{l} \cdot \mathcal{X}_{Q(l)-1}^2 + \frac{\sigma_1^2}{l} \cdot \mathcal{X}_{l-Q(l)-1}^2 \tag{4.44}$$

On peut remarquer deux comportements de V_l en fonction de l qui peuvent être distingués par :

$$\begin{cases} \text{Si } l \neq \alpha \cdot n \text{ ou } l < n_a \text{ alors } \dfrac{V_l}{M^2} \leq \frac{(q-1)^2}{q^4} \\ \text{Si } l = \alpha \cdot n \text{ et } l \geq n_a \text{ alors } \dfrac{V_l}{M^2} > \frac{(q-1)^2}{q^4} \end{cases} \tag{4.45}$$

Nous noterons $\mathcal{J}$ un ensemble défini par :

$$\mathcal{J} = \left\{ l = 2, \cdots, l_{max} | \frac{V_l}{M^2} > \frac{(q-1)^2}{q^4} \right\} \tag{4.46}$$

De ce fait, la taille des mots de code n sera identifiée en utilisant la méthode de la variance par :

$$\tilde{n} = \text{mode}(\text{diff}(\mathcal{J})) \tag{4.47}$$

Un résumé de l'algorithme d'identification basé sur le critère de la variance est présenté dans l'algorithme 4.

Algorithme 4: Algorithme basé sur le critère de la variance

Entrées : $\mathbf{r}$, M et q
Sorties : la taille identifiée de mots de code $\tilde{n}$

```
pour l = 1 à l_max faire
    Construire la matrice R̃_l de taille (M × l)
    R̃_l → T̃_l = R̃_l · A_l
    pour i = 1 à l faire
        Compter B_l(i)
    fin
    Calculer V_l
fin
Construire l'ensemble J (4.46)
Déterminer ñ (4.41)
```

Exemple 4.35.

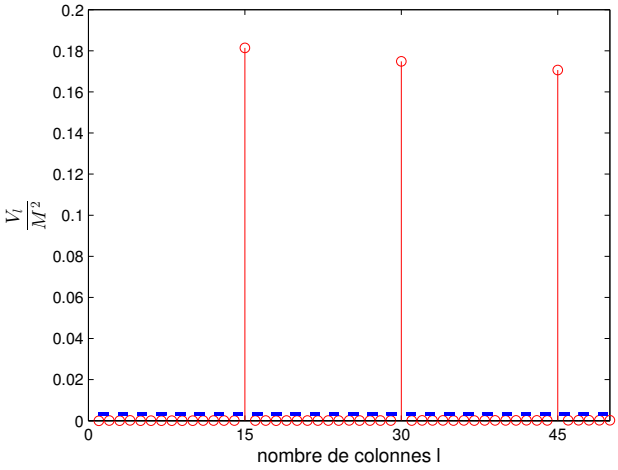
Reprenons l'exemple du code $RS(15, 11)$ dans $\text{GF}(2^4)$. Nous essayons d'identifier la taille des

mots de code par la méthode de la variance en utilisant en premier lieu $p_e = 0$, puis $p_e = 0.01$. La variance V_l normalisée par M^2, où M est fixé à 1000, est représentée sur la figure 4.9. Dans la figure 4.9(a), une probabilité d'erreur nulle (i.e. $p_e = 0$) est considérée. Pour $l \neq \alpha \cdot n$, nous pouvons vérifier que la variance V_l normalisée par M^2 est quasiment constante. Nous pouvons observer aussi des pics correspondant à $\frac{V_l}{M^2} > \frac{(q-1)^2}{q^4} = 0.0034$.

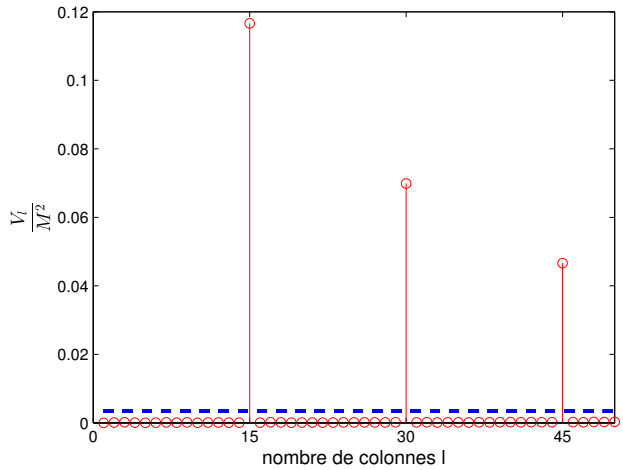
Sur la figure 6.1, la variance normalisée $\frac{V_l}{M^2}$ est représentée en fonction de l pour $p_e = 0.01$. En utilisant le définition (4.46), l'ensemble $\mathcal{J}$ est présenté dans le tableau 4.7. En utilisant l'équation (4.41), la taille identifiée de mots de code est $\tilde{n} = 15$.

$l \in \mathcal{J}$	15	30	45
diff($\mathcal{J}$)		15	15

Tableau 4.7 — Taille de matrices $\tilde{\mathbf{R}}_l$ pour $\frac{V_l}{M^2} > \frac{(q-1)^2}{q^4}$



(a) Cas $p_e = 0$



(b) Cas $p_e = 0.01$

Figure 4.9 — Variance normalisée V_l/M^2

On peut remarquer à travers l'exemple précédent que le seuil permettant de distinguer les deux comportements de V_l/M^2 n'est pas en pratique satisfaisant puisqu'il dépend du cardinal du corps q. Pour cette raison, nous choisissons un seuil plus adapté au contexte d'identification en aveugle. Il est déterminé par :

$$\max\left(V_l/M^2\right)/2$$

Ce seuil sera considéré pour l'analyse des performances des méthodes d'identification.

4.2.4 Étude comparative des méthodes présentées

L'objectif des méthodes présentées dans cette section est d'identifier en aveugle la taille des mots de code n dans un environnement bruité. Ces méthodes permettent de traiter aussi bien les codes binaires que non-binaires. Elles identifient la taille n avec une même valeur de complexité moyenne égale à $\mathcal{O}(M \cdot l_{max}^3)$. Alors, la question qui se pose est laquelle de ces méthodes devrait être utilisée. De ce fait, nous allons les comparer tout d'abord dans le cas d'une mauvaise estimation de la probabilité d'erreur. Puis, nous allons évaluer leurs performances de détection.

4.2.4.1 Impact d'une mauvaise estimation de p_e

La grande différence entre les méthodes proposées est la nécessité d'estimer la probabilité d'erreur p_e afin de déterminer le nombre de colonnes presque dépendantes $Q(l)$ pour la première méthode. En effet, les valeurs de $Q(l)$ sont calculées en utilisant le seuil γ_{opt} qui dépend des paramètres M, q et P_i. Dans le cas des méthodes de la moyenne et de la variance, il n'est pas nécessaire de connaître la probabilité P_i afin d'obtenir les valeurs de E_l et V_l. Donc, la connaissance de la probabilité d'erreur p_e n'est pas exigée pour identifier la taille des mots de code n par la deuxième et la troisième méthode.

Nous illustrons l'impact de la mauvaise estimation de la probabilité d'erreur p_e sur le calcul du nombre de colonnes presque dépendantes $Q(l)$ et sur le calcul de la moyenne E_l. Pour cela, nous considérons l'exemple précédent du code $RS(15, 11)$ dans $\mathrm{GF}(2^3)$ pour comparer les trois algorithmes proposés lorsqu'une valeur erronée de p_e est estimée. Nous supposons tout d'abord que p_e est surestimée et égale à 0.1 au lieu de la bonne valeur $p_e = 0.01$ associée au canal. Dans cette configuration, nous représentons sur la figure 6.1 le nombre de colonnes presque dépendantes en fonction de l. La valeur erronée de n égale à 1, qui peut être observée sur cette figure, est obtenue en utilisant l'équation (4.33) et les valeurs de diff($\mathcal{I}$) indiquées dans le tableau 4.8.

$l \in \mathcal{I}$	15	30	43	44	45	46	47	48	49	50
$Q(l)$	1	2	2	2	6	4	1	5	7	5
diff($\mathcal{I}$)		15	13	1	1	1	1	1	1	1

Tableau 4.8 — Nombre de colonnes presque dépendantes pour $p_e = 0.1$

Dans le cas d'une sous-estimation de la probabilité d'erreur p_e qui est estimée à $p_e = 0$, $Q(l)$ est représenté sur la figure 4.11. D'après cette figure, nous pouvons observer que l'identification de la taille des mots de code est impossible en utilisant la méthode du calcul du nombre de colonnes presque dépendantes. Cependant, la bonne taille des mots de code peut être identifiée par la deuxième et la troisième méthode même dans les cas de surestimation et de sous-estimation de p_e puisque ce paramètre n'est pas nécessaire pour calculer l'écart de la moyenne $\frac{E_l}{M} - \frac{1}{q}$ et la variance V_l.

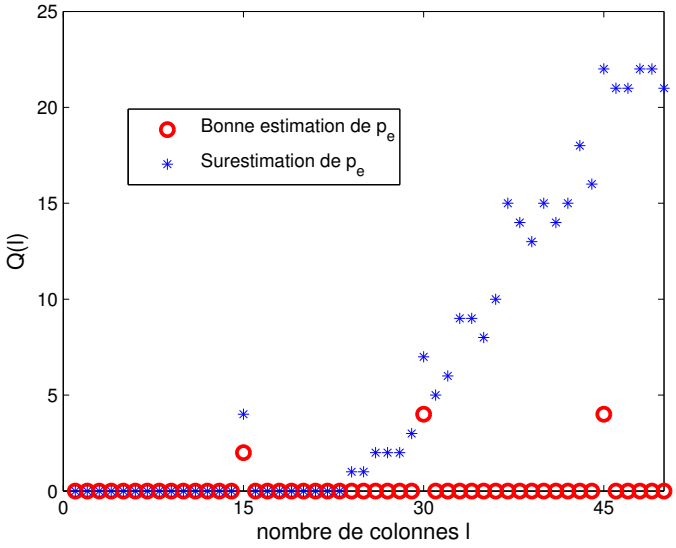


Figure 4.10 — Impact d'une surestimation de $p_e = 0.1$ au lieu de $p_e = 0.01$ sur la méthode du calcul du nombre de colonnes presque dépendantes

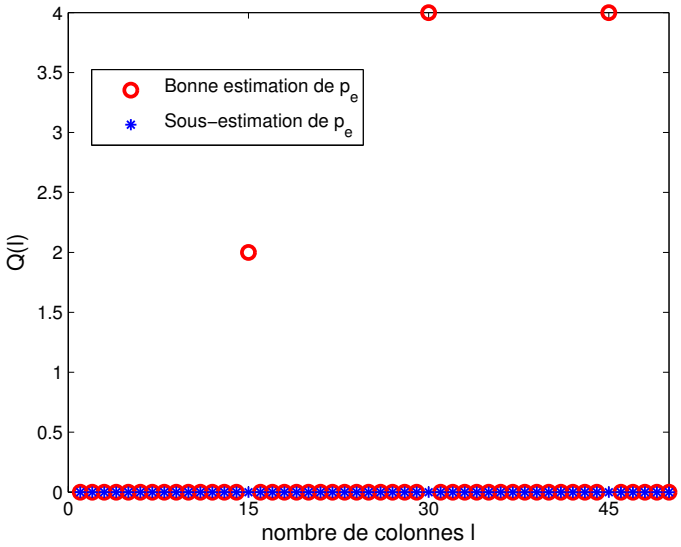


Figure 4.11 — Impact d'une sous-estimation de $p_e = 0$ au lieu de $p_e = 0.01$ sur la méthode du calcul du nombre de colonnes presque dépendantes

4.2.4.2 Analyses et performances

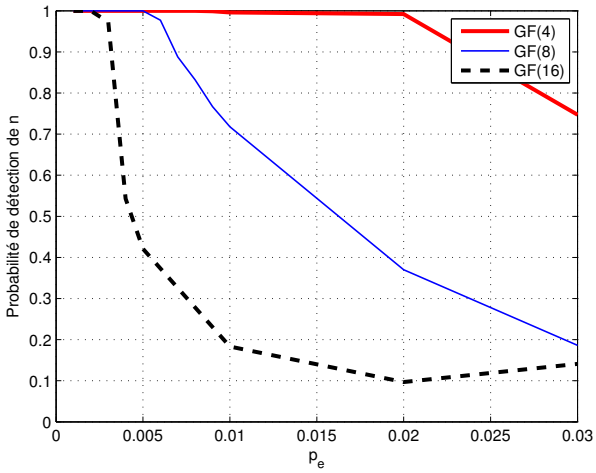
Afin d'analyser et comparer les performances de nos méthodes d'identification aveugle de la taille des mots de code, nous prenons comme critère la probabilité de détection de la bonne taille n. Dans les simulations, nos méthodes sont appliquées aux codes LDPC non-binaires qui sont devenus des candidats pour les futurs systèmes de communication. Nous supposons que la bonne probabilité d'erreur du canal p_e est estimée. Pour chaque simulation, 1000 itérations de Monte-Carlo sont réalisées, les symboles d'information étant choisis aléatoirement à chaque itération. Dans cette partie, nous nous concentrons sur :

- l'impact de l'augmentation du cardinal du corps de Galois q sur les probabilités de détection des trois méthodes proposées pour une taille n donnée,

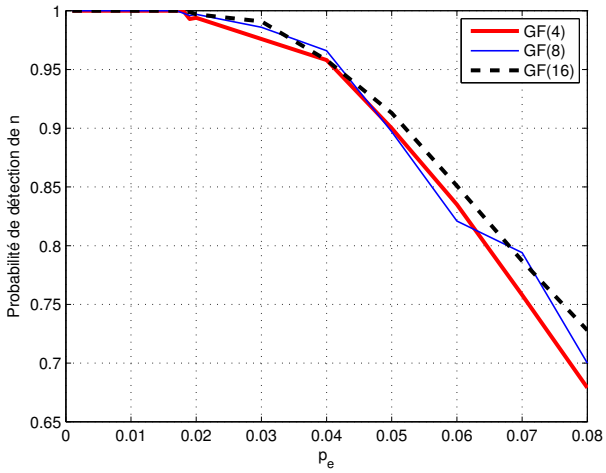
- l'impact de l'augmentation de la taille des mots de code n sur les probabilités de détection des trois méthodes proposées pour un cardinal q donné,
- la comparaison des performances des trois méthodes pour n et q donnés.

Impact de l'augmentation de q

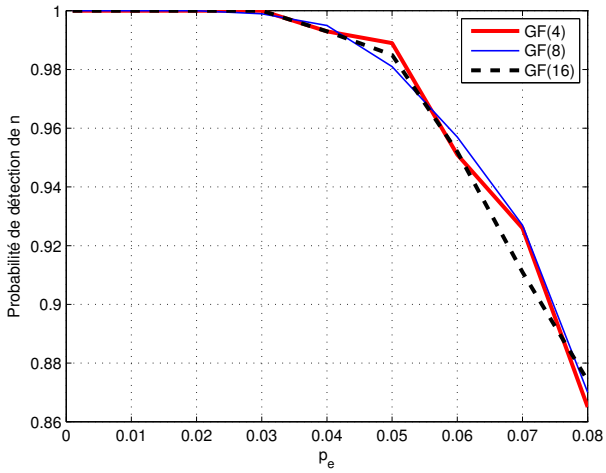
Nous considérons un code LDPC de paramètres $n = 6$ and $k = 3$, LDPC (6,3), construit sur les corps de Galois GF(q), pour $q = 4, 8, 16$. Les matrices $\tilde{\mathbf{R}}_l$ sont construites à partir des données reçues bruitées de taille $L = 30000$. Le nombre de lignes M de ces matrices est fixé à 1000 et le nombre de colonnes l varie entre 2 et 30. Pour chaque valeur de q, les trois méthodes illustrées dans la section 4.2.3 sont appliquées pour identifier en aveugle la taille des mots de code du code LDPC (6, 3) dans GF(q).



(a) Méthode basée sur le calcul du nombre de colonnes presque dépendantes



(b) Méthode basée sur la moyenne



(c) Méthode basée sur la variance

Figure 4.12 — Impact de q sur les méthodes d'identification de n pour le code LDPC (6,3)

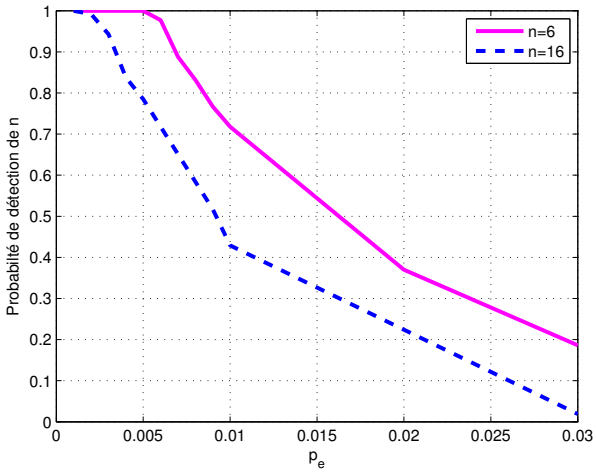
La figure 4.12(a) représente les probabilités de détection de la bonne valeur de n par la première méthode en fonction de la probabilité d'erreur p_e dans le cas de GF(4), GF(8) et GF(16). Dans la figure 4.12(b), les probabilités de détection de la deuxième méthode sont représentées et celles de

la troisième méthode sont représentées sur la figure 4.12(c). Ces figures montrent que la méthode basée sur le nombre de colonnes presque dépendantes est plus sensible à l'augmentation du cardinal du corps de Galois q que les méthodes de la moyenne et de la variance. En effet, pour $p_e = 0.01$, les probabilités de détection dans la figure 4.12(a) passent de 1 dans le cas de GF(4) à 0.7 dans le cas de GF(8) et à 0.2 dans GF(16). Cependant, les comportements des courbes de la figure 4.12(b) et la figure 4.12(c) sont quasiment semblables pour tous $q = 4$, 8, 16.

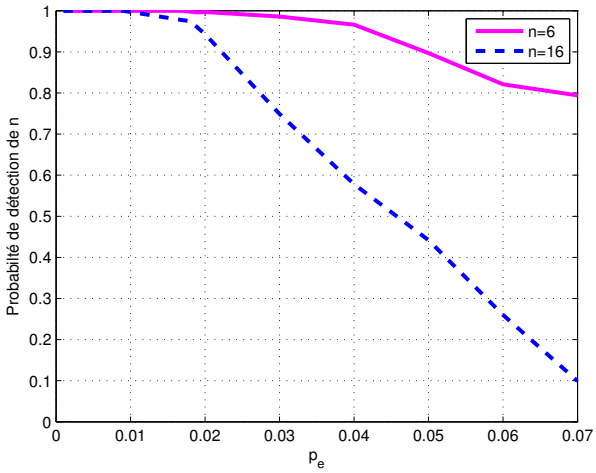
A partir de ces résultats de simulations, on peut déduire que l'augmentation du cardinal q n'a aucun effet sur les performances de détection de la deuxième et de la troisième méthode.

Impact de l'augmentation de n

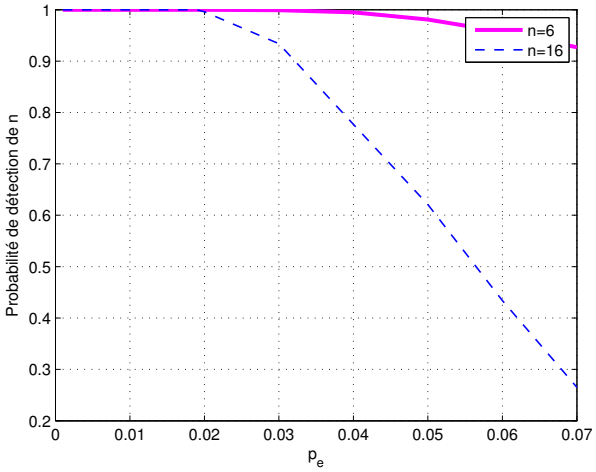
Pour évaluer les performances de détection de nos méthodes d'identification aveugle, l'impact de l'augmentation de la taille des mots de code est étudié. Dans nos simulations, nous considérons deux codes LDPC dans GF(2^3), LDPC (6, 3) et LDPC (16, 8).



(a) Méthode basée sur le calcul du nombre de colonnes presque dépendantes



(b) Méthode basée sur la moyenne



(c) Méthode basée sur la variance

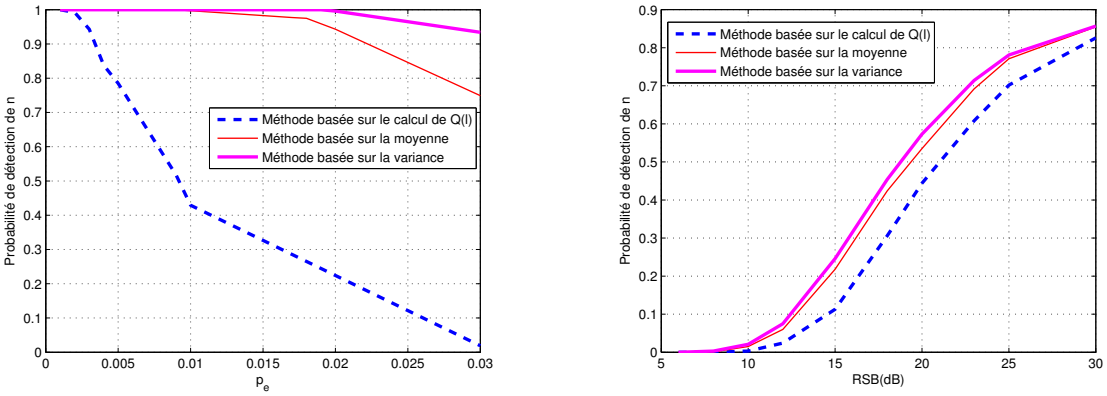
Figure 4.13 — Impact de l'augmentation de n sur les méthodes d'identification pour des codes dans GF(2^3)

Les probabilités de détection de n par la méthode basée sur le calcul du nombre de colonnes presque dépendantes $Q(l)$ en fonction de p_e sont représentées sur la figure 4.13(a). La figure 4.13(b) et la figure 4.13(c) montrent respectivement les probabilités de détection de la méthode basée sur le critère de la moyenne et celles de la méthode basée sur la variance. On peut constater que l'augmentation de la taille des mots de code entraîne une dégradation des performances pour les trois méthodes proposées, mais pour des valeurs de p_e différentes. En effet, pour $p_e = 0.01$, les probabilités de détection de la première méthode passent de 0.71 dans le cas du code LDPC(6, 3) à 0.43 dans le cas du code LDPC(16, 8). Pour la méthode de la moyenne, les probabilités de détection diminuent de 0.96 à 0.57 dans le cas de $p_e = 0.04$. Pour $p_e = 0.05$, la probabilité de détection de la méthode de la variance passe de 0.98 à 0.6210.

Il est possible d'améliorer les performances de nos méthodes d'identification en augmentant le nombre de données reçues.

Comparaison de performances pour q et n données

Comparons maintenant les performances obtenues par nos méthodes d'identification pour le code LDPC (16, 8) dans GF(2^3) en considérant un canal 8-aire symétrique (premier canal) et un canal de Rayleigh à trajets multiples associé à un canal BBAG (deuxième canal). Dans le cas du deuxième canal, les symboles d'information sont transmis via une modulation PAM (Pulse Amplitude Modulation) d'ordre 8. Afin de compenser et réduire les interférences entre les symboles provoquées par la propagation multi-trajets, un égaliseur linéaire MMSE de longueur 20 a été utilisé. La probabilité d'erreur moyenne, p_e, en sortie de cet égaliseur peut être obtenue par l'expression (10.2-62 dans [Pro01]). Cette probabilité est prise en compte à l'entrée de la première méthode d'identification de n.



(a) Canal 8-aire symétrique

(b) Canal de Rayleigh multi-trajet avec $L_{path} = 4$

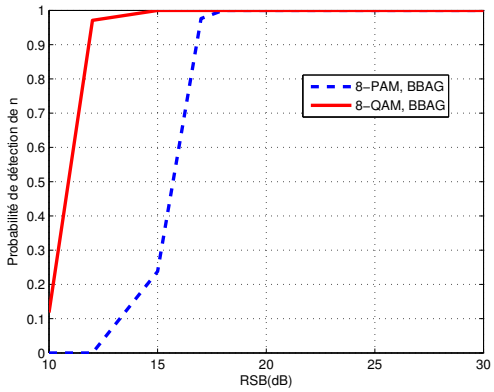
Figure 4.14 — Comparaison des performances entre nos trois méthodes pour un code LDPC(16,8) et une modulation 8-PAM

Les probabilités de détection de nos méthodes dans le cas des deux canaux sont représentées sur la figure 4.14. Nous pouvons observer que la troisième méthode améliore de manière significative les performances de détection comparée à la première méthode. En effet, nous pouvons observer sur la figure 4.14(a) que la probabilité de détection de la première méthode est égale à 0.43 pour $p_e = 0.01$, alors que, pour la même p_e, nous obtenons une probabilité de détection égale à 1 avec la troisième méthode. On peut aussi remarquer que la deuxième méthode a de meilleurs performances que la première méthode et des performances moins bonnes par rapport à la méthodes de la variance. Pour $p_e = 0.03$, la probabilité de détection est de 0.0190 dans le cas de la première méthode, de 0.7490 dans le cas de la méthode de la moyenne et de 0.9340 avec la méthode de la variance. Sur la figure

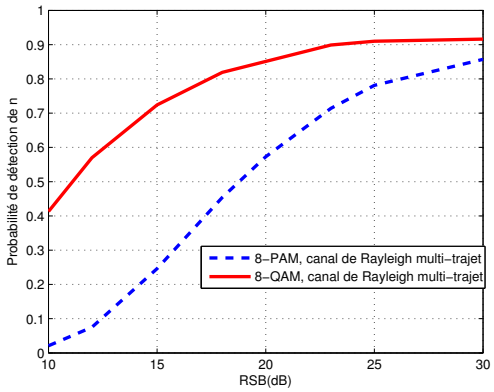
4.14(b), nous représentons les probabilités de détection en fonction du RSB(dB) dans le cas du canal quasi-statique de Rayleigh à trajets multiples avec un nombre de trajets $L_{path} = 4$. Comme le cas du canal 8-aire symétrique, nous pouvons observer de meilleurs performances de détection avec les méthodes de la variance et de la moyenne comparées à la méthode basée sur le calcul du nombre de colonnes presque dépendantes. Par contre, les courbes de performances de la méthode de la variance et de la moyenne sont quasiment identiques. Pour un RSB= 25 dB, la probabilité de détection est égale à 0.78 pour la troisième méthode, 0.77 pour la deuxième méthode et 0.70 pour la première méthode.

On peut déduire d'après l'étude des performances de détection de nos méthodes d'identification que la méthode de la variance offre de meilleurs performances dans le cas du canal q-aire symétrique et le cas d'un canal réel comme le canal de Rayleigh à trajets multiples.

Dans ce paragraphe, nous évaluons les performances de détection de la méthode basée sur la variance lorsqu'une modulation PAM ou QAM d'ordre 8 (8-QAM et 8-PAM) est utilisée pour transmettre les symboles codés par un LDPC (16, 8) dans GF(2^3).



(a) Canal BBAG



(b) Canal de Rayleigh multi-trajets avec $L_{path} = 4$

Figure 4.15 — Performances de détection de la méthode de la variance pour LDPC (16, 8)

Sur la figure 4.15, nous représentons une comparaison des performances de la méthode basée sur la variance en utilisant les modulations 8-PAM ou 8-QAM dans le cas des canaux BBAG et Rayleigh multi-trajets avec un nombre de trajets $L_{path} = 4$. Sur la figure 4.15(a), une comparaison des performances de détection est représentée dans le cas du canal BBAG. Nous pouvons observer que notre méthode d'identification pour la modulation 8-QAM offre de meilleurs performances que pour la modulation 8-PAM pour un RSB< 18 dB. Le gain entre les deux modulation est de 5 dB. En revanche, pour un RSB >18 dB, les performances sont similaires et la probabilité de détection est égale à 1.

Pour obtenir les probabilités de détection présentées dans la figure 4.15(b), les symboles issus des modulations 8-PAM ou 8-QAM sont transmis sur le canal quasi-statique de Rayleigh à trajets multiples avec un nombre de trajets $L_{path} = 4$, puis les symboles reçus sont traités par un égaliseur linéaire MMSE de longueur 20. Nous constatons que, dans le cas de la modulation 8-QAM, notre méthode d'identification offre de meilleurs performances que dans le cas de la modulation 8-PAM. Dans ce cas, un gain de 7 dB est apporté.

Nous avons choisi de comparer nos méthodes d'identification sur la figure 4.14(b) dans le pire cas de modulation 8-PAM. Notre objectif était de montrer que la méthode basée sur le critère de la variance possède de meilleures performances, même dans le cas de la modulation PAM.

Par la suite, nous présentons des méthodes d'identification d'une base du code dual pour des codes binaires et non-binaires en bloc en supposant que la taille des mots de code est identifiée à

l'aide de la méthode de la variance présentée dans cette section.

4.3 Algorithmes d'identification aveugle à décision ferme d'une base du code dual

Dans cette section, nous mettrons en place des algorithmes qui permettent d'identifier en aveugle une base du code dual pour des codes correcteurs d'erreurs en bloc binaires et non-binaires dans le cas d'un environnement bruité. Ces algorithmes sont basés sur l'utilisation des informations issues d'un modulateur à décision ferme, aussi appelées informations "hard".

4.3.1 Codes binaires

Dans cette partie, nous décrivons le principe de la méthode classique d'identification d'une base de code dual pour des codes binaires lorsque la séquence de mots de code est transmise sur un canal binaire symétrique (BSC) de probabilité d'erreur p_e. Nous supposons une synchronisation parfaite et la taille des mots de code n connue.

Nous rappelons qu'un code dual est défini par :

$$C^{\perp} = \{\mathbf{h} | \forall \mathbf{c} \in C,\ \mathbf{c} \cdot \mathbf{h}^T = 0\} \tag{4.48}$$

où le vecteur $\mathbf{h}$ correspond à une relation de parité de taille n. Le vecteur $\mathbf{c} = \begin{pmatrix} c_1 & c_2 & \cdots & c_n \end{pmatrix}$ est un mot de code de taille n. Tout vecteur $\mathbf{h}$ doit vérifier la relation définie par l'équation (4.48) pour tous les mots de code. Afin de réaliser l'opération de décodage, il est nécessaire d'identifier $n-k$ vecteurs de taille n appartenant à $C^{\perp}$. L'idée est de réorganiser la séquence des mots de code reçus entachés d'erreurs de longueur L sous forme d'une matrice $\tilde{\mathbf{R}}_n$ de taille $(M \times n)$ où la taille des mots de code n correspond au nombre de colonnes. La matrice de contrôle de parité $\mathbf{H}$ est de taille $((n-k) \times n)$. Comme mentionné dans la section 3.5, nous pouvons identifier en aveugle $\mathbf{H}$ dans un environnement de transmission non-bruitée en résolvant le système décrit par :

$$\mathbf{R}_n \cdot \mathbf{H}^T = 0 \tag{4.49}$$

Cependant, l'équation (4.49) ne sera pas satisfaite lorsque les mots de code reçus sont corrompus. Dans ce cas, la méthode proposée dans [MGB11] peut être utilisée pour identifier la matrice $\mathbf{H}$ ou sa matrice équivalente. Le principe de cette méthode est de transformer la matrice $\tilde{\mathbf{R}}_n$ en une matrice triangulaire, notée $\tilde{\mathbf{T}}_n$, de taille $(M \times n)$ en utilisant l'élimination de Gauss dans GF(2). Le résultat de cette transformation peut être décrite par un produit matriciel sous la forme $\tilde{\mathbf{T}}_n = \tilde{\mathbf{R}}_n \cdot \mathbf{A}_n$, où $\mathbf{A}_n$ est une matrice de taille $(n \times n)$ correspondant aux combinaisons de colonnes. Dans la prochaine étape de la méthode, nous comptons le nombre de 0 ou de 1 ($B_n(i)$) dans la i-ème colonne de la matrice $\tilde{\mathbf{T}}_n$. Dans [Mar09], les auteurs ont montré que la variable $B_n(i)$ suit deux lois différentes en fonction de l'appartenance ou non du vecteur candidat $\mathbf{a}_i$, la i-ème colonne de $\mathbf{A}_n$, au code dual. Ils ont proposé la règle de décision suivante :

$$\begin{cases} \mathbf{a}_i \in \mathcal{C}^{\perp} \text{ si } B_n(i) > \frac{M}{2} \cdot \gamma_{opt} \\ \mathbf{a}_i \notin \mathcal{C} \text{ si } B_n(i) \leq \frac{M}{2} \cdot \gamma_{opt} \end{cases} \tag{4.50}$$

où γ_{opt} est un seuil optimal qui peut être déterminé par l'équation (4.4). Pour calculer cette probabilité, il est nécessaire de connaître la probabilité d'erreur p_e et le poids de Hamming de $\mathbf{a}_i$. Si p_e est connue par le récepteur, nous sommes alors capable de déterminer le seuil γ_{opt} pour chaque vecteur candidat $\mathbf{a}_i$ et de décider s'il appartient ou non à la base, notée $\mathcal{D}$, du code dual $\mathcal{C}^{\perp}$. La méthode

d'identification d'une base $\mathcal{D}$ du code dual pour des codes binaire est résumée dans l'algorithme 5.

Algorithme 5: Algorithme d'identification d'une base $\mathcal{D}$ pour des codes binaires à décision ferme

Entrées : $\tilde{\mathbf{r}}$, n, M et p_e
Sorties : Base identifiée $\mathcal{D}$

Construire la matrice $\tilde{\mathbf{R}}_n$ de taille $(M \times n)$
$\mathbf{R}_n \rightarrow \tilde{\mathbf{T}}_n = \tilde{\mathbf{R}}_n \cdot \mathbf{A}_n$
pour $i = 1$ *à* n **faire**
  Calculer P_i (4.2)
  Calculer γ_{opt} (4.4)
  si $B_l(i) > \frac{M}{2} \cdot \gamma_{opt}$ **alors**
    $\mathcal{D} \leftarrow \mathcal{D} \cup \mathbf{a}_i$
  fin
fin

Pour améliorer la probabilité de détection de $n-k$ équations de parité, un processus itératif a été proposé dans [MGB11]. Le principe de ce processus est de permuter aléatoirement les lignes de la matrice $\tilde{\mathbf{R}}_n$ à chaque itération afin de réduire la probabilité d'obtenir des pivots erronés durant l'élimination de Gauss, ce qui s'avère catastrophique pour la suite du calcul. Notre objectif par la suite est de présenter certaines améliorations à l'opération de permutation en utilisant la décision souple sur les données reçues. Avant d'expliquer nos améliorations dans le cas de la décision souple, nous présentons notre généralisation de la méthode d'identification d'une base du code dual au cas des codes en bloc non-binaires pour une modulation à décision ferme.

4.3.2 Codes non-binaires

En considérant un canal q-aire symétrique de probabilité d'erreur p_e, à partir de la connaissance de p_e et de la taille de mots de code n qui peut être identifiée par l'un de nos algorithmes proposés dans la section 4.2.3, nous pouvons estimer une base du code dual d'un code en bloc non-binaire. Comme dans le cas binaire, la première étape de la méthode d'identification d'une base du code dual est de construire la matrice $\tilde{\mathbf{R}}_n$ à partir des symboles reçus entachés d'erreurs et issus d'un modulateur d'ordre q à décision ferme, c'est-à-dire les symboles reçus sont des éléments du corps GF(q).

Nous avons montré dans la sous-section 4.2.3.1 que la variable $B_l(i)$ possède aussi deux comportements différents en fonction de l'appartenance du vecteur $\mathbf{a}_i^{(l)}$ au code dual. D'après les différentes définitions que nos avons vues, les deux comportement sont délimités comme suit :

$$\begin{cases} \text{Si } B_l(i) > \frac{M}{q} \cdot \eta_{opt} \text{ alors } \mathbf{a}_i^{(l)} \in \mathcal{C}^{\perp} \\ \text{Si } B_l(i) \leq \frac{M}{q} \cdot \eta_{opt} \text{ alors } \mathbf{a}_i^{(l)} \notin \mathcal{C}^{\perp} \end{cases}$$

Nous avons montré que le seuil optimal η_{opt} dépend de la probabilité P_i donnée par l'équation (4.12). D'après cette équation, cette probabilité varie en fonction de la probabilité d'erreur p_e, du poids de Hamming de $\mathbf{a}_i^{(l)}$ et du cardinal du corps q. Donc, pour un vecteur candidat $\mathbf{a}_i^{(l)}$, nous calculons le seuil optimal qui lui correspond après avoir déterminé la probabilité P_i qui lui est associée. Les différentes étapes représentées dans l'algorithme 6 permettent d'obtenir en sortie un ensemble $\mathcal{D}$ de relations de parité.

Algorithme 6: Algorithme d'identification d'une base $\mathcal{D}$ pour des codes non-binaires à décision ferme

Entrées : $\tilde{\mathbf{r}}$, n, M, p_e et q
Sorties : Base identifiée $\mathcal{D}$

Construire la matrice $\tilde{\mathbf{R}}_n$ de taille $(M \times n)$
$\mathbf{R}_n \rightarrow \tilde{\mathbf{T}}_n = \tilde{\mathbf{R}}_n \cdot \mathbf{A}_n$
pour $i = 1$ *à* n **faire**
  Calculer P_i (4.12)
  Calculer γ_{opt} (4.28)
  si $B_l(i) > \frac{M}{q} \cdot \gamma_{opt}$ **alors**
    $\mathcal{D} \leftarrow \mathcal{D} \cup \mathbf{a}_i$
  fin
fin

4.4 Algorithmes d'identification aveugle à décision souple d'une base du code dual

Dans cette section, nous étendons les travaux précédents de Sicot et al. [SH05] qui ont proposé une méthode basée sur les propriétés d'algèbre linéaire afin de reconnaître en aveugle la taille d'entrelacement et la taille des mots de code à partir des données reçues codées par un code en bloc. Cette méthode a été revisitée par Barbier et al. dans [BL09] pour récupérer les paramètres d'un code en bloc linéaire et par Marazin et al. dans [MGB09b, MGB11] pour identifier en aveugle une base du code dual pour les codes convolutifs. Son principe est d'identifier les paramètres d'un code binaire à partir des matrices construites à partir des données obtenues en sortie d'un démodulateur à décision ferme. Dans la section précédente, nous avons généralisé cette méthode au cas des codes non-binaires. Afin d'améliorer la probabilité de détection de la base du code dual, un algorithme itératif qui consiste à permuter aléatoirement les lignes des matrices à chaque itération a été proposé dans [MGB11].

La plupart des méthodes publiées pour identifier une base du code dual sont basées sur l'utilisation d'un démodulateur à décision ferme. Dans cette section, nous proposons une méthode qui tire profit des caractéristiques de l'information souple et du rôle de l'algorithme de décodage à décision souple afin d'avoir de meilleures performances d'estimation d'une base du code dual. En utilisant l'algorithme itératif de décodage souple, nous pouvons corriger les symboles erronés afin d'améliorer la probabilité de détection d'une équation de parité. De ce fait, l'algorithme de décodage prend en entrée les équations de parité identifiées et fournit en sortie l'information souple des symboles corrigés. Le principe général de fonctionnement de l'algorithme itératif d'identification est illustré sur la figure 4.16.

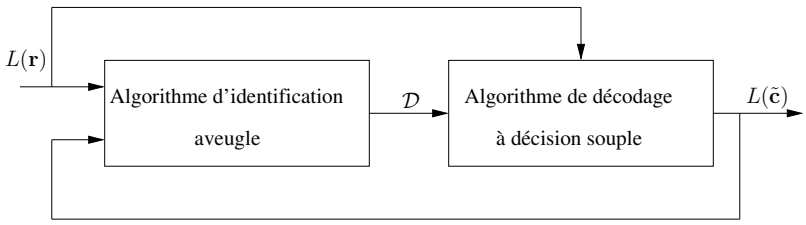


Figure 4.16 — Algorithme itératif d'identification

De nos jours, la décision souple est surtout utilisée par les algorithmes de décodage les plus efficaces, comme l'algorithme de Viterbi et les algorithmes de décodage des codes LDPC, car elle

fournit une information sur la fiabilité de la décision. En outre, l'utilisation de la décision souple dans les algorithmes de décodage offre généralement de meilleures performances en termes de taux d'erreur binaire (TEB) par rapport à l'utilisation des informations fermes. Dans le cadre de notre étude, nous profitons de cette caractéristique afin d'améliorer l'algorithme d'identification aveugle en supposant que la taille des mots d'information k et la taille des mots de code sont connues. Nous supposons également que la synchronisation des mots de code est parfaite, c'est-à-dire qu'on sait où se trouve le début de chaque mot de code.

4.4.1 Principe

Dans cette partie, nous présentons les différentes étapes d'amélioration de l'algorithme classique d'identification illustré dans la sous section 4.3.1. Dans la première étape, nous proposons deux critères basés sur l'utilisation de l'information souple issue du démodulateur afin de ranger les mots de code suivant leur propreté sous forme d'une matrice, c'est-à-dire que les mots de code les moins entachés d'erreurs sont placés dans les premières lignes de cette matrice. Dans la deuxième étape, nous considérons un algorithme de décodage à décision souple pour les codes LDPC qui est basé sur la propagation de croyance afin de corriger les mots de code reçus. Nous allons montrer que l'utilisation de l'algorithme de décodage peut améliorer les performances de détection de l'algorithme d'identification classique. Cet algorithme de décodage a besoin en entrée de $n-k$ relations de parité. En revanche, l'algorithme d'identification peut identifier plus que ce nombre de relations. Dans cette situation, notre idée est de modifier l'algorithme de décodage afin de prendre en compte la probabilité de fiabilité d'une relation de parité estimée par tous les mots de code. Dans le cas d'un excès de relations estimées, nous utilisons cette probabilité de fiabilité afin d'éliminer les relations les moins fiables et garder uniquement les $n-k$ relations les plus fiables. Les probabilités de fiabilité correspondant aux $n-k$ relations de parité vont être utilisées par le nouveau algorithme de décodage.

4.4.2 Améliorations par tri des mots de code les moins entachés d'erreurs

La première idée d'amélioration s'inspire de [SHB09]. Le principe de cette idée est basé sur l'utilisation de l'information souple des bits reçus. En effet, à travers les valeurs souples des bits reçus obtenus par un démodulateur à décision souple, nous pouvons extraire les mots de code les moins erronés dans les données reçues pour construire la matrice $\tilde{\mathbf{R}}_n$. Par conséquent, l'élimination des mots de code les plus corrompus nous permet d'améliorer la probabilité de détection des équations de contrôle de parité d'un code $\mathcal{C}$.

Les informations souples (LRV) $L(\mathbf{r})$ sont réorganisées sous la forme d'une matrice, notée $\hat{\mathbf{R}}_n$, de taille $(M \times n)$ construite de la même façon que $\mathbf{R}_n$. Par la suite, la fiabilité de chaque mot de code (ou chaque ligne de $\hat{\mathbf{R}}_n$) est calculée. En fait, la fiabilité de la décision est définie par la valeur absolue de l'information souple (LRV). De ce fait, nous utilisons la fonction $F(j)$ qui estime la fiabilité de la j-ème ligne de $\hat{\mathbf{R}}_n$ (j-ème mot de code) :

$$F(j) = \frac{\sum_{i=1}^{n} |\hat{R}_n(j,i)|}{n} \tag{4.51}$$

où $\hat{R}_n(j,i)$ est un élément de la j-ème ligne et de la i-ème colonne de $\hat{\mathbf{R}}_n$. Les valeurs de $F(j)$ sont triées par ordre décroissant afin de permuter les lignes selon les mots de code les plus fiables. Par la suite, les lignes de la nouvelle matrice sont ordonnées en fonction du nombre d'erreurs par ligne ou par mot de code. Une estimation du nombre de bits fiables dans la j-ème ligne de la nouvelle matrice $\hat{\mathbf{R}}_n$ est déterminée par :

$$Z(j) = card\left\{ i \in \{1, \cdots, n\} \, |\hat{R}_n(j,i)| > S \right\} \tag{4.52}$$

où S est le seuil de la fiabilité de la décision souple. En effet, la décision du bit est fiable lorsque la valeur absolue de l'information souple du bit reçu est supérieure au seuil S. Afin de décider la valeur du bit transmis, ce seuil est normalement fixé à 0. Cependant, avec $S = 0$, une mauvaise décision peut être réalisée avec une certaine probabilité. Un choix optimal du seuil est indispensable dans notre algorithme proposé afin d'éliminer les mots de code qui contiennent le nombre maximum de bits erronés. Nous avons choisi la valeur du seuil S tel que la probabilité de mauvaise décision soit égale à 0.0033.

Notons τ le point du seuil à partir duquel la décision ferme est fiable. Ce point et la distribution de la probabilité de deux signaux transmis sur un canal Gaussien sont représentés sur la figure 4.17. La valeur de τ peut être calculée par :

$$\tau = 1 - \sigma \cdot \exp(1) \tag{4.53}$$

Le lecteur intéressé pourra se référer à l'annexe G où une démonstration détaillée de l'équation (4.53) est présentée. Par conséquent, le seuil de la décision souple S peut être déterminé par :

$$S = |L(\tau)| = \frac{2}{\sigma^2} \cdot |1 - \sigma \cdot \exp(1)| \tag{4.54}$$

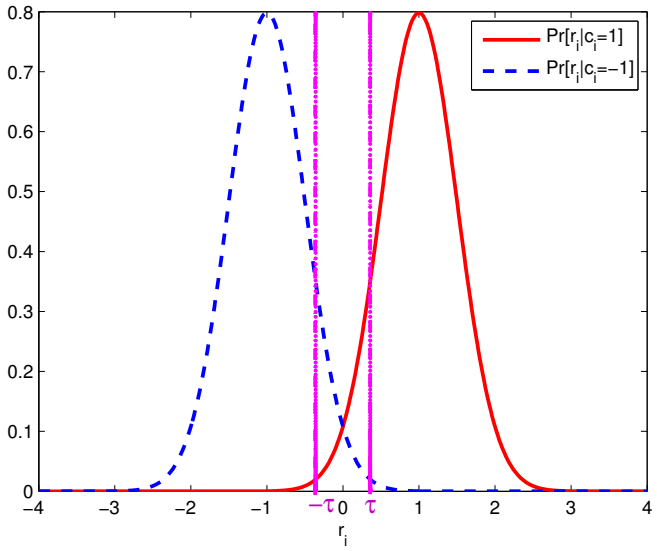


Figure 4.17 — Distributions de deux signaux transmis sur un canal Gaussien

Une fois que les opérations de permutation de lignes sont réalisées, l'algorithme d'identification classique décrit dans la sous-section 4.3.1 peut être appliqué en utilisant la nouvelle matrice $\tilde{\mathbf{R}}_n$ obtenue après les opérations de permutation et de démodulation à décision ferme. Cette dernière est appliquée afin d'obtenir des éléments binaires dans la matrice $\tilde{\mathbf{R}}_n$.

4.4.3 Algorithme conjoint d'identification et de décodage à décision souple

4.4.3.1 Algorithme de décodage classique du code LDPC binaire

L'objectif du décodeur est d'associer le message le plus probable au message émis $\mathbf{c}$ qui est reçu erroné à cause du canal de transmission. Le message décodé peut être estimé en utilisant l'une des deux techniques qui dépendent du format du message issu du démodulateur. La première technique est le décodage ferme qui utilise en entrée des valeurs binaires suite à la décision ferme

du démodulateur. La deuxième technique est le décodage souple qui utilise des valeurs réelles obtenues par la démodulation à décision souple. Dans le cadre de cette thèse, nous nous intéressons à l'algorithme de décodage à décision souple pour les codes LDPC.

Dans notre cas, le décodage d'un code LDPC est basé sur un algorithme itératif appelé algorithme à propagation de croyance. A chaque itération, les noeuds de variable et les noeuds de parité échangent les messages d'information qui transportent essentiellement les informations extrinsèques calculées par les noeuds de parité et les informations à priori estimées par les noeuds de variable. Le décodage à entrée souple cherche un mot de code $\mathbf{c}$ ayant des bits c_j, pour $j \in [\![1, n]\!]$, qui permettent de maximiser la probabilité :

$$Pr\left[c_j|\mathbf{r}, \text{toutes les relations de parité incluant } c_j \text{ sont satisfaites}\right]$$

Cette probabilité correspond à la probabilité a posteriori pour un bit sachant que toutes les relations de parité impliquant ce bit sont satisfaites. Le critère utilisé dans cet algorithme de décodage est dit critère du Maximum A Posteriori (MAP). Afin de simplifier les calculs et réduire la complexité du décodage, le logarithme du rapport de vraisemblance (LRV) de c_j, noté $\gamma(c_j|\mathbf{r})$, est utilisé :

$$\gamma(c_j|\mathbf{r}) = \log\left(\frac{Pr[c_j = 1|\mathbf{r}]}{Pr[c_j = 0|\mathbf{r}]}\right) \tag{4.55}$$

En appliquant la loi de Bayes, l'équation (4.55) devient :

$$\gamma(c_j|\mathbf{r}) = \underbrace{\log\left(\frac{Pr[r_j|c_j = 1]}{Pr[r_j|c_j = 0]}\right)}_{\text{LRV intrinsèque}} + \underbrace{\log\left(\frac{Pr[c_j = 1|\{r_s, s \neq j\}]}{Pr[c_j = 0|\{r_s, s \neq j\}]}\right)}_{\text{LRV extrinsèque}} \tag{4.56}$$

D'après cette équation, on peut remarquer que le LRV de c_j qui correspond au LRV a posteriori est décomposé en un LRV intrinsèque et un LRV extrinsèque que nous allons calculer par la suite.

Considérons un canal BBAG de variance σ^2. L'information intrinsèque qui provient de l'observation (i.e les mots de code reçus) peut alors s'écrire sous la forme :

$$\log\left(\frac{Pr[r_j|c_j = 1]}{Pr[r_j|c_j = 0]}\right) = L_c \cdot r_j \tag{4.57}$$

où $L_c = 2 \cdot \sqrt{E_s}/\sigma^2$ représente la fiabilité du canal avec E_s l'énergie d'un symbole transmis.

L'information extrinsèque est déterminée par les informations provenant des équations de parité. Notons $\mathcal{M}_j = \{i|H_{i,j} = 1\}$ l'ensemble des équations de parité vérifiées par le bit c_j et $\mathcal{N}_i = \{j|H_{i,j} = 1\}$ l'ensemble des bits qui participent à l'équation de parité z_i. Une équation de parité z_i peut s'écrire :

$$z_i = \sum_{s \in \mathcal{N}_i} c_s \tag{4.58}$$

L'ensemble des bits qui participent à l'équation z_i autres que le bit c_j est noté par $\mathcal{N}_{i,j} = \mathcal{N}_i \setminus \{j\}$. Nous notons également $\mathcal{M}_{j,i} = \mathcal{M}_j \setminus \{i\}$ l'ensemble des équations de parité autre que z_i auquel le bit c_j participe. Soit $z_{i,j}$ l'équation définie par :

$$z_{i,j} = \sum_{s \in \mathcal{N}_{i,j}} c_s \tag{4.59}$$

Alors, l'équation $z_{i,j} + c_j = 0$ nous permet de calculer le LRV extrinsèque. En effet, si $c_j = 0$ ou $c_j = 1$ alors $z_{i,j} = 0$ ou $z_{i,j} = 1$. Dans ce cas, la probabilité que l'équation z_i soit vérifiée par c_j est

supposée égale à 1. De ce fait, le LRV extrinsèque, noté LRV_{ext}, peut s'écrire :

$$LRV_{ext} = \log\left(\frac{Pr[z_{i,j} = 1, \forall i \in \mathcal{M}_j | \{r_s, s \neq j\}]}{Pr[z_{i,j} = 0, \forall i \in \mathcal{M}_j | \{r_s, s \neq j\}]}\right) \tag{4.60}$$

Supposons que le graphe de Tanner décrivant la structure du code ne contient aucun cycle. Alors, les bits de l'équation $z_{i,j}$ sont indépendants de ceux de l'équation $z_{i',j}$, avec $i \neq i'$. Par conséquent, l'équation (4.60) peut s'écrire :

$$LRV_{ext} = \sum_{i \in \mathcal{M}_j} \log\left(\frac{Pr[z_{i,j} = 1 | \{r_s, s \neq j\}]}{Pr[z_{i,j} = 0 | \{r_s, s \neq j\}]}\right) \tag{4.61}$$

Pour simplifier, notons le logarithme du rapport de vraisemblance :

$$\gamma\left(z_{i,j} = 1 | \{r_s, s \neq j\}\right) = \log\left(\frac{Pr[z_{i,j} = 1 | \{r_s, s \neq j\}]}{Pr[z_{i,j} = 0 | \{r_s, s \neq j\}]}\right) \tag{4.62}$$

Alors, l'équation (4.61) peut s'écrire :

$$\begin{aligned} LRV_{ext} &= \sum_{i \in \mathcal{M}_j} \gamma\left(z_{i,j} = 1 | \{r_s, s \neq j\}\right) && (4.63) \\ &= \sum_{i \in \mathcal{M}_j} \gamma\left(\sum_{k \in \mathcal{N}_{i,j}} c_k | \{r_s, s \neq j\}\right) && (4.64) \\ &= -2 \cdot \sum_{i \in \mathcal{M}_j} \tanh^{-1}\left(\prod_{k \in \mathcal{N}_{i,j}} \left(-\frac{\lambda(c_k | \{r_s, s \neq j\})}{2}\right)\right) && (4.65) \end{aligned}$$

Nous définissons :

$$\eta_{i,j} = -2 \cdot \tanh^{-1}\left(\prod_{k \in \mathcal{N}_{i,j}} \left(-\frac{\lambda(c_k | \{r_s, s \neq j\})}{2}\right)\right) \tag{4.66}$$

l'information envoyée par le noeud de parité i au noeud de variable j. Alors, l'équation (4.55) devient :

$$\gamma(c_j | \mathbf{r}) = L_c \cdot r_j + \sum_{i \in \mathcal{M}_j} \eta_{i,j} \tag{4.67}$$

On peut définir le LRV $\gamma(c_j|\mathbf{r})$ comme l'information envoyée par le noeud de variable j au noeud de parité i. Nous vérifions donc que l'algorithme itératif du décodage est basé sur l'échange d'informations entre les noeuds de variables et les noeuds de parité. En effet, à chaque itération, les équations (4.66) et (4.67) sont mises à jour. Les étapes de cette approche sont décrites dans l'algorithme 7.

Algorithme 7: Algorithme classique de décodage du code LDPC binaire

Entrées : un mot de code bruité $\mathbf{r}$, le nombre maximal d'itérations *iter* et L_c
Sorties : γ, le mot de code corrigé $\tilde{\mathbf{c}}$

Initialisation : Pour tous les mots de code bruités $\mathbf{r}$: $\eta_{i,j} = 0$, $\forall(i,j)$
$\gamma_j = L_c \cdot r_j$
Itérer jusqu'à iter : Pour tous les mots $\mathbf{r}$:
Mise à jour du noeud de parité : Pour chaque (i,j), avec $H(i,j) = 1$, calculer :

$$\eta_{i,j} = -2 \cdot \tanh^{-1}\left(\prod_{s \in \mathcal{N}_{i,j}} \left(-\frac{\gamma_s - \eta_{i,s}}{2}\right)\right) \tag{4.68}$$

Mise à jour du noeud de variable : $\forall j$, calculer :

$$\gamma_j = L_c \cdot r_j + \sum_{i \in \mathcal{M}_j} \eta_{i,j} \tag{4.69}$$

Terminaison : Prendre la décision : si $\gamma_j > 0$ alors $\tilde{c}_j = 1$ sinon $\tilde{c}_j = 0$

4.4.3.2 Algorithme modifié de décodage utilisant les probabilités de fiabilité des relations de parité identifiées

Le principe général de notre algorithme itératif d'identification d'une base du code dual est illustré sur la figure 4.16. Afin d'améliorer les performances de cet algorithme, nous proposons de modifier l'algorithme classique de décodage d'un code LDPC présenté dans le paragraphe précédent afin qu'il prenne en compte les fiabilités des équations de parité estimées par notre méthode d'identification présentée dans l'algorithme 5 .

L'algorithme de décodage illustré dans la sous-section 4.4.3.1 présente deux opérations de mise à jour des LRVs. Dans la première opération, on met à jour les noeuds de parité définis par le logarithme du rapport de vraisemblance extrinsèque LRV_{ext}. Dans ce cas, l'équation de parité z_i dans laquelle le bit c_j intervient est supposée vérifiée avec une probabilité égale à 1. Dans le cadre de notre étude, le décodeur prend en entrée des équations de parité estimées et il donne en sortie des bits corrigés qui vont être utilisés par la suite pour une nouvelle identification des équations de parité. De ce fait, notre algorithme d'identification associé avec les opérations de permutation devient un algorithme itératif qui exploite la capacité de correction du décodeur afin d'améliorer ses performances de détection de relations de parité.

Nous supposons que les équations identifiées sont vérifiées avec une probabilité p_i (i.e $Pr[z_i = 0] = p_i$). On souhaite alors modifier l'algorithme classique de décodage en prenant en compte la probabilité p_i. On aura :

$$\begin{aligned} Pr\left[c_j = 1 | Pr[z_i = 0] = p_i\right] &= p_i \cdot Pr[z_{i,j} = 1] + (1 - p_i) \cdot Pr[z_{i,j} = 0] \\ Pr\left[c_j = 0 | Pr[z_i = 0] = p_i\right] &= p_i \cdot Pr[z_{i,j} = 0] + (1 - p_i) \cdot Pr[z_{i,j} = 1] \end{aligned} \tag{4.70}$$

Dans ce cas, le LRV extrinsèque modifié, noté LRV'_{ext}, est donné par :

$$\begin{aligned} LRV'_{ext} &= \log\left(\frac{\prod_{i\in\mathcal{M}_j} p_i \cdot Pr\left[z_{i,j}=1|\{r_s, s\neq j\}\right] + \prod_{i\in\mathcal{M}_j}(1-p_i)\cdot Pr\left[z_{i,j}=0|\{r_s, s\neq j\}\right]}{\prod_{i\in\mathcal{M}_j} p_i \cdot Pr\left[z_{i,j}=0|r_s, s\neq j\right] + \prod_{i\in\mathcal{M}_j}(1-p_i)\cdot Pr\left[z_{i,j}=1|\{r_s, s\neq j\}\right]}\right) \\ &= \log\left(\frac{\prod_{i\in\mathcal{M}_j} p_i \cdot \dfrac{\prod_{i\in\mathcal{M}_j} Pr\left[z_{i,j}=1|\{r_s, s\neq j\}\right]}{\prod_{i\in\mathcal{M}_j} Pr\left[z_{i,j}=0|\{r_s, s\neq j\}\right]} + \prod_{i\in\mathcal{M}_j}(1-p_i)}{\prod_{i\in\mathcal{M}_j} p_i + \prod_{i\in\mathcal{M}_j}(1-p_i)\cdot \dfrac{\prod_{i\in\mathcal{M}_j} Pr\left[z_{i,j}=1|\{r_s, s\neq j\}\right]}{\prod_{i\in\mathcal{M}_j} Pr\left[z_{i,j}=0|\{r_s, s\neq j\}\right]}}\right) \end{aligned} \tag{4.71}$$

Nous notons $\lambda_j = \log\left(\dfrac{\prod_{i\in\mathcal{M}_j} Pr\left[z_{i,j}=1|\{r_s, s\neq j\}\right]}{\prod_{i\in\mathcal{M}_j} Pr\left[z_{i,j}=0|\{r_s, s\neq j\}\right]}\right)$. On a :

$$\begin{aligned} \lambda_j &= \sum_{i\in\mathcal{M}_j} \log\left(\frac{Pr\left[z_{i,j}=1|\{r_s, s\neq j\}\right]}{Pr\left[z_{i,j}=0|\{r_s, s\neq j\}\right]}\right) \\ &= -2\cdot\sum_{i\in\mathcal{M}_j} \tanh^{-1}\left(\prod_{k\in\mathcal{N}_{i,j}} \tanh\left(-\frac{\gamma(c_k|\{r_s, s\neq j\})}{2}\right)\right) \qquad (4.72) \\ &= \sum_{i\in\mathcal{M}_j} \eta_{i,j} \qquad (4.73) \end{aligned}$$

où $\eta_{i,j}$ est donné par :

$$\eta_{i,j} = -2\cdot\tanh^{-1}\left(\prod_{k\in\mathcal{N}_{i,j}} \tanh\left(-\frac{\gamma(c_k|\{r_s, s\neq j\})}{2}\right)\right) \tag{4.74}$$

Afin de mettre à jour le noeud de parité, l'équation (4.74) devient :

$$\eta_{i,j} = -2\cdot tanh^{-1}\left(\prod_{k\in\mathcal{N}_{i,j}} \tanh\left(-\frac{\gamma(c_k|\{r_s, s\neq j\}) - \eta_{i,k}}{2}\right)\right) \tag{4.75}$$

Ainsi, l'équation (4.71) peut s'écrire :

$$LRV'_{ext} = \eta_j = \log\left(\frac{\exp(\lambda_j)\cdot\prod_{i\in\mathcal{M}_j} p_i + \prod_{i\in\mathcal{M}_j}(1-p_i)}{\prod_{i\in\mathcal{M}_j} p_i + \exp(\lambda_j)\cdot\prod_{i\in\mathcal{M}_j}(1-p_i)}\right) \tag{4.76}$$

La mise à jour du noeud de variable est réalisée par le calcul du :

$$\gamma_j = \gamma(c_j|\mathbf{r}) = L_c\cdot r_j + \eta_j \tag{4.77}$$

Un mot de code, noté $\mathbf{c}^{(t)} = \left(c_1^{(t)}, c_2^{(t)}\cdots c_n^{(t)}\right), \forall t\in\{1,\cdots,M\}$, avec M le nombre de mots de code, vérifie l'équation z_i avec une probabilité $\tilde{p}_i^{(t)}$ qui est donnée par :

$$\tilde{p}_i^{(t)} = \frac{1+\prod_{s\in\mathcal{N}_i}\left(1-2\cdot Pr\left[c_s^{(t)}=1|\mathbf{r}\right]\right)}{2} \tag{4.78}$$

avec :

$$Pr\left[c_s^{(t)} = 1|\mathbf{r}\right] = \frac{\exp\left(\gamma(c_s^{(t)}|\mathbf{r})\right)}{1 + \exp\left(\gamma(c_s^{(t)}|\mathbf{r})\right)} \tag{4.79}$$

La probabilité que l'équation z_i soit vraie en utilisant toutes les informations dont on dispose sur cette équation exceptée celle fournie par le mot $\mathbf{c}^{(t)}$ est donnée par :

$$p_i^{(t)} = \frac{\sum_{s \neq t} \tilde{p}_m^{(s)}}{M - 1} \tag{4.80}$$

La probabilité a posteriori que l'équation z_i soit vérifiée par tous les mots de code est donnée par :

$$p_i = \frac{\sum_{t=1}^{M} \tilde{p}_i^{(t)}}{M} \tag{4.81}$$

La technique de décodage fondée sur les probabilités de fiabilité des équations de parité estimées est illustrée par l'algorithme 8. A l'entrée de cet algorithme, on dispose de données reçues $\mathbf{r}$ qui sont réorganisées sous forme d'une matrice, notée $\tilde{\mathbf{R}}$, de taille $(M' \times n)$, avec $M' = \lfloor L/n \rfloor$ où chaque ligne représente un mot de code bruité. Chaque mot de code de cette matrice va être corrigée en utilisant les équations de parité identifiées dont le nombre, noté nb_{Eq}, est inférieur ou égal à $n - k$. Ces équations de parité estimées forment une matrice, notée $\mathbf{D}$, de taille $(nb_{Eq} \times n)$ qui peut être considérée comme une matrice de parité dans notre contexte. En plus de la matrice $\mathbf{D}$, l'algorithme de décodage a aussi besoin en entrée de connaître la probabilité de fiabilité, $p_i^{(t)}$ de chaque équation de parité, i, où $i \in [\![1, nb_{Eq}]\!]$, pour chaque mot de code, t, où $t \in [\![1, M']\!]$, calculée en sortie de l'algorithme d'identification de $\mathbf{D}$. Nous construisons avec les probabilités calculées, $p_i^{(t)}$, une matrice, notée $\mathbf{p}$, de taille $(M' \times nb_{Eq})$, où l'indice de ligne est t et l'indice de colonne est i. L'algorithme de décodage donne en sortie les informations a posteriori sur les mots de code corrigés sous forme d'une matrice, notée γ, de taille $(M' \times n)$ ainsi que la matrice des mots de code corrigés sous forme binaire, notée $\tilde{\mathbf{C}}$, et la matrice des probabilités de fiabilité $\mathbf{p}$ mise à jour.

Algorithme 8: Nouvel algorithme de décodage du code LDPC binaire

Entrées : $\tilde{\mathbf{R}}$, $\mathbf{D}$, $\mathbf{p}$, Lc et le nombre maximal d'itérations *iter*
Sorties : γ, $\tilde{\mathbf{C}}$ et $\mathbf{p}$

Initialisation : Pour tout $i \in [\![1, nb_{Eq}]\!]$, $j \in [\![1, n]\!]$ et $t \in [\![1, M']\!]$:
$\eta_{i,j}^{(t)} = 0$
$\gamma_j^{(t)} = L_c \cdot \tilde{r}_j^{(t)}$

Itérer jusqu'à *iter* **:**
1. Pour tout $i \in [\![1, nb_{Eq}]\!]$ et $t \in [\![1, M']\!]$,
calculer $p_i^{(t)}$ (4.80)

2. **Mise à jour du noeud de parité :** Pour chaque (i, j), avec $D(i, j) = 1$, et pour $t \in [\![1, M']\!]$, calculer $\eta_j^{(t)}$ en utilisant les équations (4.72), (4.75) et (4.76)

3. **Mise à jour du noeud de variable :** $\forall j \in [\![1, n]\!]$ et $t \in [\![1, M']\!]$,
calculer $\gamma_j^{(t)}$ en utilisant l'équation (4.77)

4. **Mise à jour des probabilités de fiabilité :** $\forall i \in [\![1, nb_{Eq}]\!]$ et $t \in [\![1, M']\!]$,
calculer $\tilde{p}_i^{(t)}$ en utilisant les équations (4.78)et (4.79)

5. **Terminaison :**
Prendre la décision pour tous les mots de code : si $\gamma_j^{(t)} > 0$ alors $\tilde{c}_j^{(t)} = 1$ sinon $\tilde{c}_j^{(t)} = 0$
Pour toute équation z_i, calculer p_i en utilisant l'équation (4.81)

Critère d'arrêt : Si $\mathbf{D} \cdot \tilde{\mathbf{C}}^T = \mathbf{0}$, alors arrêter les itérations sinon retourner à l'étape 1.

4.4.3.3 Nouvelle méthode itérative d'identification d'une base de code dual

Nous mentionnons que nous nous intéressons dans cette section aux codes en bloc linéaires binaires. Le problème qui se pose ici est le suivant : la trame de bits codés puis modulés $\mathbf{r}$ observée en sortie d'un canal BBAG de probabilité d'erreur p_e, donnée par l'équation (2.5), est entachée d'erreurs. Cette trame est une concaténation de mots de code bruités. Après avoir démodulé cette trame, le rôle du récepteur est de la corriger à l'aide d'un algorithme de décodage. Mais, le bloc de décodage ne peut pas réaliser la correction de la trame sans connaître la matrice de contrôle de parité du code utilisée à l'émission ou une matrice équivalente. Dans le cadre de cette thèse, notre objectif est d'identifier une matrice équivalente à la matrice de parité du codeur qui permet aussi de faire la correction des données bruitées. En effet, l'utilisation d'une matrice équivalente de la matrice de contrôle de parité permet de corriger la trame des mots de code bruités mais ne permet pas de récupérer les mots d'information. Pour cela, le code $\mathcal{C}$ et le code dual $\mathcal{C}^{\perp}$ peuvent être vus comme deux espaces vectoriels. Nous allons donc chercher une base du code dual comme espace vectoriel. Cette problématique a été déjà traité dans [Val00, SHB09, Mar09] pour des faibles probabilités d'erreurs et en utilisant des démodulateurs à décision ferme.

Nous proposons ici un algorithme itératif d'identification qui exploite les informations souples sur la fiabilité des bits afin d'éliminer les mots de code les plus entachés d'erreurs. De ce fait, pour identifier une base du code dual, nous utilisons seulement les mots de code les moins entachés d'erreurs. La base identifiée qui peut comporter un nombre insuffisant d'équations de parité sera ensuite utilisée par le décodeur afin de corriger, même partiellement, les mots de code reçus. L'information souple obtenue en sortie de l'algorithme de décodage sera réutilisée à l'itération suivante pour dé-

tecter les mots de code les moins entachés d'erreurs qui vont servir à l'identification d'une autre base. Celle-ci sera concaténée à la base de l'itération précédente pour pouvoir compléter l'ancienne base incomplète en vérifiant que les relations de parité des deux bases sont indépendantes. La base construite va être utilisée par le décodeur afin de corriger plus d'erreurs. Ce procédé est réitéré et cela jusqu'à la dernière itération. Le processus itératif correspondant qui prend en compte les probabilités de fiabilité des relations de parité identifiées est illustré comme suit :

Initialisation :

- Nous construisons la matrice $\hat{\mathbf{R}}$ de taille $(M' \times n)$ à partir des informations souples $L(\tilde{\mathbf{r}})$ des données reçues.
- Nous initialisons $\hat{\mathbf{R}}_n = \hat{\mathbf{R}}$.
- Nous initialisons la matrice de contrôle de parité estimée $\mathbf{D}$ et la matrice des probabilités de fiabilité $\tilde{\mathbf{p}}$ par des matrices vides :
  - $\mathbf{D} = (\cdot)$;
  - $\tilde{\mathbf{p}} = (\cdot)$
- Nous initialisons l'itération, notée it, à 1.

Processus itératif :

- Nous appliquons les opérations de permutation de lignes, illustrées dans la sous-section 4.4.2, sur la matrice $\hat{\mathbf{R}}_n$ afin de réorganiser ses lignes suivant les mots de code les plus fiables et les moins entachés d'erreurs.
- Nous utilisons un démodulateur à décision ferme, où la décision est définie par l'équation (2.8), pour convertir les éléments souples de la matrice $\hat{\mathbf{R}}_n$ en des éléments binaires. A partir de la matrice binaire obtenue, nous prenons les M premières lignes correspondant aux M mots de code les plus fiables. Dans ce cas, nous obtenons une nouvelle matrice, notée $\tilde{\mathbf{R}}_n$, de taille $(M \times n)$.
- En utilisant la matrice $\tilde{\mathbf{R}}_n$ et la probabilité d'erreur p_e qui peut être déterminée par l'équation (2.5), nous appliquons la technique d'identification aveugle présentée dans la sous section 4.3.1 pour estimer une base du code dual, notée $\mathbf{D}^{(it)}$.
- Nous concaténons la base identifiée à l'itération actuelle $\mathbf{D}^{(it)}$ avec la base de l'itération précédente, $\mathbf{D}^{(it-1)}$, $\mathbf{D} \leftarrow \mathbf{D}^{(it-1)} \cup \mathbf{D}^{(it)}$, pour combler l'insuffisance des équations de parité. Dans le cas où les équations sont complètes, la base identifiée à l'itération actuelle contiendra des colonnes dépendantes de la base de l'itération précédente. Donc, nous pouvons les éliminer et ne garder que les colonnes indépendantes dans $\mathbf{D}$.
- Pour des probabilités d'erreur élevées, la probabilité de fausse détection d'une équation de parité augmente au niveau de l'algorithme d'identification. Celui-ci engendre un excès d'équations de parité identifiées, c'est-à-dire qu'on identifie plus de $n-k$ équations. C'est pourquoi, nous proposons de calculer, à chaque itération et après l'identification, les probabilités de fiabilité $\tilde{\mathbf{p}}^{(it)}$ des équations de parités estimées pour tous les mots de code. En fait, ces probabilités nous permettent de connaître les $n-k$ équations de parité les plus fiables et d'éliminer les moins fiables. Pour ce faire, nous vérifions tout d'abord si le nombre d'équations de parité identifiées à l'itération actuelle est supérieur à celui de l'itération précédente. Si c'est le cas, on peut connaître les nouvelles équations ajoutées à cette itération et calculer leurs probabilités de fiabilité $\tilde{\mathbf{p}}^{(it)}$ à l'aide des équations (4.78) et (4.79). Puis, nous concaténons les probabilités qui sont déjà calculées à l'itération précédente $\tilde{\mathbf{p}}^{(it-1)}$ avec les nouvelles : $\tilde{\mathbf{p}} \leftarrow \tilde{\mathbf{p}}^{(it-1)} \cup \tilde{\mathbf{p}}^{(it)}$. S'il n'y a pas de nouvelles équations de parité identifiées, alors $\tilde{\mathbf{p}} \leftarrow \tilde{\mathbf{p}}^{(it-1)}$.

- Nous vérifions si $\tilde{\mathbf{p}}$ contient plus de $n-k$ équations de parité. Si c'est le cas, alors nous trions les équations identifiées en fonction de leurs fiabilités et nous gardons uniquement les $n-k$ équations qui ont les probabilités de fiabilité les plus élevées. En fait, notre algorithme de décodage peut accepter l'insuffisance d'équations de parité qui permettent de corriger partiellement les mots de code reçus. Mais, il ne peut pas accepter l'excès d'équations car il peut dans ce cas engendrer des erreurs supplémentaires dans les mots de code et dégrader les performances de détection de l'algorithme d'identification.

- Nous appliquons maintenant la nouvelle technique de décodage présentée dans l'algorithme 8 qui prend en entrée la matrice $\tilde{\mathbf{R}}$ de taille $(M' \times n)$ construite à partir des mots de code bruités. Le rôle de cette technique est de corriger ces mots de code à l'aide de la matrice de parité identifiée $\mathbf{D}$ en prenant en compte les probabilités de fiabilité de chaque équation de parité pour chaque mot de code, $\tilde{\mathbf{p}}$. Cette technique permet également d'augmenter les probabilités de fiabilité des bonnes équations de parité qui convergent vers 1 lorsqu'on augmente le nombre d'itérations de notre processus itératif. La mise à jour de ces probabilités est réalisée pendant l'opération de décodage. Les probabilités obtenues à la sortie du décodeur, $\tilde{\mathbf{p}}$, ainsi que la matrice des informations souples des mots de code corrigés, γ, seront réutilisées à l'itération suivante.

- Nous affectons $\mathbf{D}$ à $\mathbf{D}^{(it)}$ et $\tilde{\mathbf{p}}$ à $\tilde{\mathbf{p}}^{(it)}$: $\mathbf{D}^{(it)} \leftarrow \mathbf{D}$ et $\tilde{\mathbf{p}}^{(it)} \leftarrow \tilde{\mathbf{p}}$.

- Nous prenons un nouveau jeu de données, c'est-à-dire la matrice des informations souples $\hat{\mathbf{R}}_n$ est construite à partir de la matrice γ en excluant les M première lignes à chaque itération.

Terminaison :

- Nous obtenons en sortie, après it_{max} itérations, la base du code dual identifiée, $\mathbf{D}^{(it_{max})}$ qui contient les équations de parité les plus fiables ainsi que leurs probabilités de fiabilité $\tilde{\mathbf{p}}^{(it_{max})}$.

Notre processus itératif d'identification aveugle d'une base du code dual est résumé sur la figure 4.18.

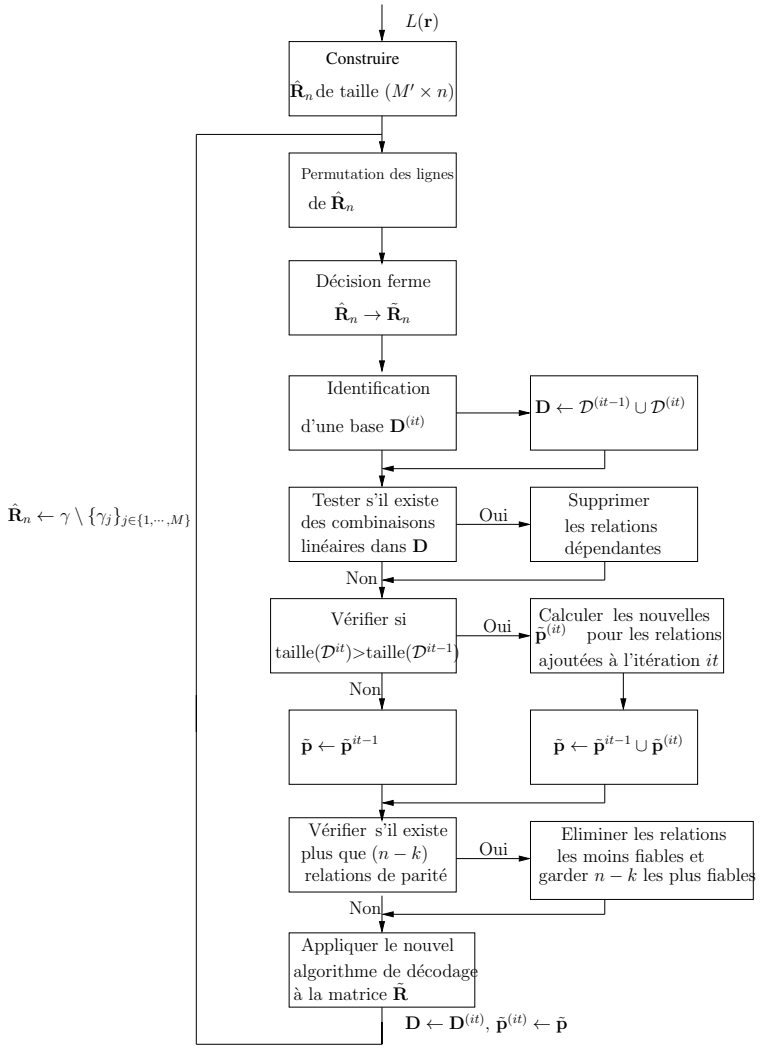


Figure 4.18 — Schéma du processus itératif d'identification d'une base du code dual binaire

4.4.4 Analyse des algorithmes

Les résultats des simulations dans cette section ont pour objectif d'évaluer les performances en termes de probabilité de détection, noté P_{det}, de nos algorithmes d'identification d'une base du code dual pour les codes binaires et non-binaires. Dans notre contexte, la détection inclut l'identification de $n-k$ équations de parité. Nos simulations mettent en évidence l'intérêt du processus itératif en termes de performances lorsqu'on utilise des informations souples issues du décodeur pour identifier en aveugle une base du code dual. Tout d'abord, quelques rappels sont nécessaires à la compréhension des résultats et des courbes de performances obtenues :

- Le train des mots de code reçus est synchronisé.
- Les paramètres du codeur n et k sont supposés connus à la réception.
- Le train de symboles ou de bits est composé d'au minimum $it_{max} \cdot n \cdot M$ symboles ou bits.
- Une modulation BPSK est utilisée pour les codes binaires.
- Une modulation QAM d'ordre 2^m est utilisée pour les codes dans GF(2^m).
- Un canal BBAG est considéré,

- Les paramètres du corps de Galois GF(2^m) utilisés à l'émission sont supposés connus à la réception.

Dans notre étude, nous avons pris en compte des débits proposés par les standards actuels comme l'UMTS ou LTE. En considérant le standard LTE, le débit maximal théorique de la première catégorie de ce standard peut atteindre 10 Mbps pour une largeur de bande de 20 MHz. Dans nos simulations, nous considérons une trame de symboles transmis de longueur 20000 bits ou symboles. Ainsi, nos algorithmes d'identification peuvent être adaptés dans un contexte réel. Pour chaque simulation, 1000 itérations de Monte-Carlo sont réalisées où des symboles d'information sont choisis aléatoirement à chaque itération. Dans cette partie, nous nous intéressons à des codes LDPC de paramètres suivants :

- Le code LDPC(6,3) dans GF(2) de matrice de contrôle de parité :

$$\mathbf{H} = \begin{pmatrix} 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \tag{4.82}$$

- Le code LDPC(12,6) dans GF(2) de matrice de contrôle de parité :

$$\mathbf{H} = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix} \tag{4.83}$$

- Le code LDPC(6,3) dans GF(2^4) de matrice de contrôle de parité :

$$\mathbf{H} = \begin{pmatrix} 0 & 15 & 7 & 0 & 6 & 0 \\ 7 & 0 & 0 & 12 & 0 & 4 \\ 1 & 0 & 1 & 0 & 0 & 8 \end{pmatrix} \tag{4.84}$$

4.4.4.1 Probabilités de détection

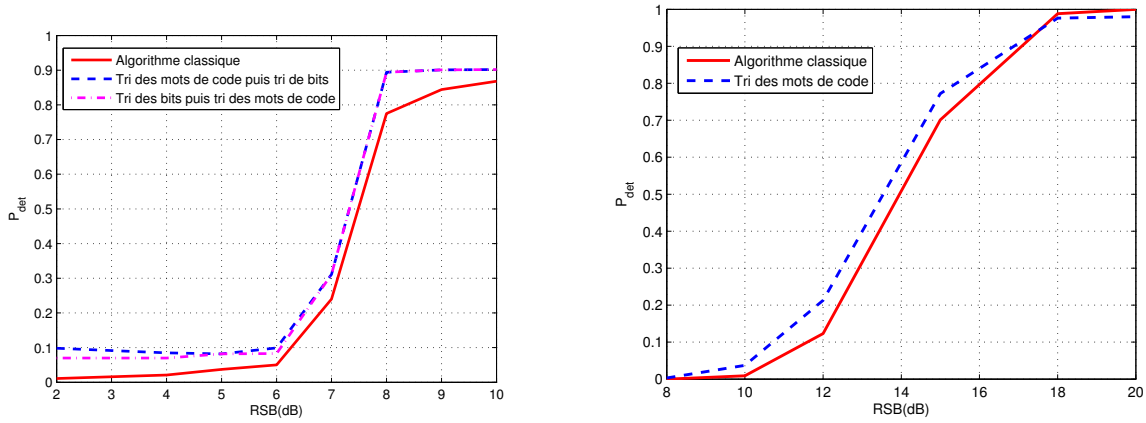
Dans le but d'analyser les performances de détection de nos algorithmes d'identification d'une base du code dual, nous définissons trois probabilités :

- Probabilité de détection P_{det} : la probabilité d'identifier $n-k$ équations de parité appartenant au code dual.
- Probabilité de fausse alarme P_{fa} : la probabilité d'identifier $n-k$ fausses équations de parité.
- Probabilité de non-détection P_{nd} : la probabilité de n'identifier aucune équation de parité ou d'identifier un nombre d'équations inférieur à $n-k$.

Afin de comparer les performances de détection entre l'algorithme classique à décision ferme et l'algorithme amélioré par le tri des mots de code les moins entachés d'erreurs, nous représentons sur la figure 4.19 les probabilités de détection de ces algorithmes en fonction du RSB(dB) pour le code LDPC(6,3) dans GF(2) et le code LDPC(6,3) dans GF(2^4).

Pour le code LDPC(6,3) dans GF(2), la probabilité de détection en fonction du rapport RSB(dB) est représentée sur la figure 4.19(a). Sur cette figure, nous avons également représenté les courbes

de performances de l'algorithme amélioré par le tri dans deux cas. Le premier cas correspond à trier les bits les moins bruités en premier lieu, c'est-à-dire avant de trier des mots de code. Dans le deuxième cas, il s'agit de trier des mots de code les moins entachés d'erreurs avant de trier les bits. L'objectif est de détecter à partir de ces deux courbes le cas qui a des meilleurs performances afin de le prendre en compte pour l'amélioration de la capacité de détection de notre algorithme. D'après la figure 4.19(a), nous remarquons que les deux courbes sont quasiment identiques. On peut en déduire que les deux cas ont quasiment les mêmes probabilités de détection. Par la suite, nous considérons le deuxième cas qui correspond à trier les mots de code avant les bits.



(a) Code LDPC(6,3) dans GF(2)

(b) Code LDPC(6,3) dans GF(2^4)

Figure 4.19 — Comparaison des performances entre les algorithmes classique et de tri des mots de code

En comparant les performances de détection de l'algorithme classique à l'algorithme amélioré par le tri, nous observons de bonnes performances pour l'algorithme amélioré qui apporte un gain de 0.3 dB pour $P_{det} = 0.2$.

Pour le code LDPC(6,3) dans GF(2^4), la probabilité de détection en fonction du rapport RSB(dB) est représentée sur la figure 4.19(b). Nous pouvons également observer de meilleurs performances pour l'algorithme amélioré par le tri des mots de code les moins entachés d'erreurs par rapport à l'algorithme classique pour de faibles valeurs de RSB(dB). Pour des RSB(dB)$\geq$ 18 dB, les deux courbes de performances sont similaires.

A travers la figure 4.19, nous avons mis en évidence l'amélioration des performances de détection de l'algorithme d'identification lorsque les méthodes de tri illustrées dans la sous section 4.4.2 sont utilisées.

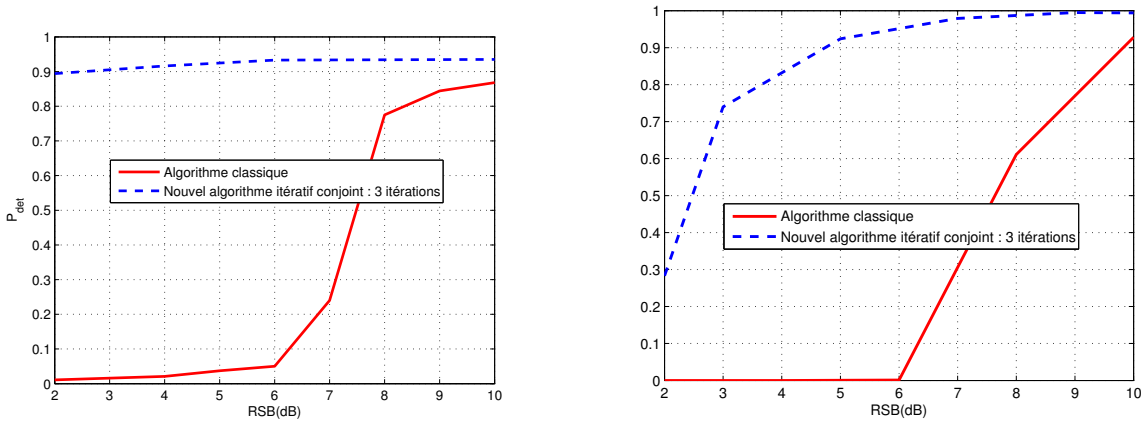
Comparons maintenant les performances de détection de l'algorithme classique et de l'algorithme conjoint d'identification et de décodage à décision souple basé sur le tri des mots de code les moins bruités. Pour cela, nous avons réalisé 3 itérations pour l'algorithme d'identification conjoint. A chaque itération, un algorithme modifiée de décodage LDPC à 50 itérations a été utilisé pour corriger les mots de code. Nous avons représenté sur la figure 4.20 la probabilité de détection de ces deux algorithmes d'identification en fonction du RSB(dB) pour le code LDPC(6,3) dans GF(2) et le code LDPC(12,6) dans GF(2).

Sur la figure 4.20(a), nous avons représenté la probabilité de détection P_{det} en fonction du RSB(dB) pour les deux algorithmes d'identification pour le cas du LDPC(6,3) dans GF(2). Nous constatons que le nouvel algorithme itératif d'identification présente de meilleurs performances de détection comparé à l'algorithme classique. Cet algorithme présente un gain significatif qui dépasse 10 dB. En effet, même pour des valeurs faibles du rapport RSB(dB) qui sont comprises entre 2 et

6 dB, les probabilités de détection du nouvel algorithme sont supérieures ou égales à 0.9, alors que les probabilités de détection de l'algorithme classique sont inférieures à 0.1.

Sur la figure 4.20(b), nous représentons les probabilités de détection pour les deux algorithmes d'identification dans le cas du code LDPC(12,6) dans GF(2). Nous remarquons également des améliorations significatives des performances de détection lorsque l'algorithme d'identification itératif est utilisé. Cet algorithme a apporté un gain de 5.7 dB pour une $P_{det} = 0.7$.

Nous venons de montrer que les performances de détection de l'algorithme d'identification classique sont largement améliorées par le processus itératif qui est basé sur l'utilisation conjointe de l'algorithme d'identification et du décodage itératif à décision souple prenant en compte les probabilités de fiabilités des équations de parité identifiées.



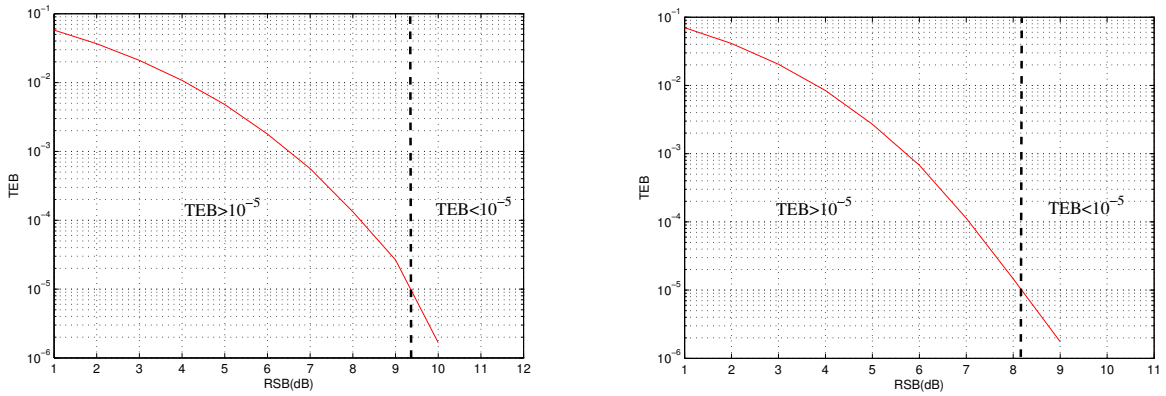
(a) Code LDPC(6,3) dans GF(2)

(b) Code LDPC(12,6) dans GF(2)

Figure 4.20 — Comparaison des performances entre l'algorithme classique et le nouvel algorithme

Afin d'évaluer la pertinence de nos résultats, une comparaison entre les performances de détection et le pouvoir de correction du code est souhaitable. Le critère considéré pour évaluer le pouvoir de correction du code est le taux d'erreur binaire TEB obtenu après décodage des mots de code bruités. Pour garantir une qualité de service raisonnable dans les standards actuels, les taux d'erreur binaires acceptables doivent être inférieurs à 10^{-5}.

Dans nos simulations, nous avons considéré un décodage LDPC à décision souple et un canal BBAG. Sur la figure 4.21, nous avons représenté les TEB en fonction du RSB(dB) pour les trois codes : LDPC(6,3) dans GF(2), LDPC(12,6) dans GF(2) et LDPC(6,3) dans GF(2^4). Les TEB ont été calculés pour un RSB(dB) variant de 1 à 10 pour les codes binaires et de 1 à 14 pour les codes non-binaires.



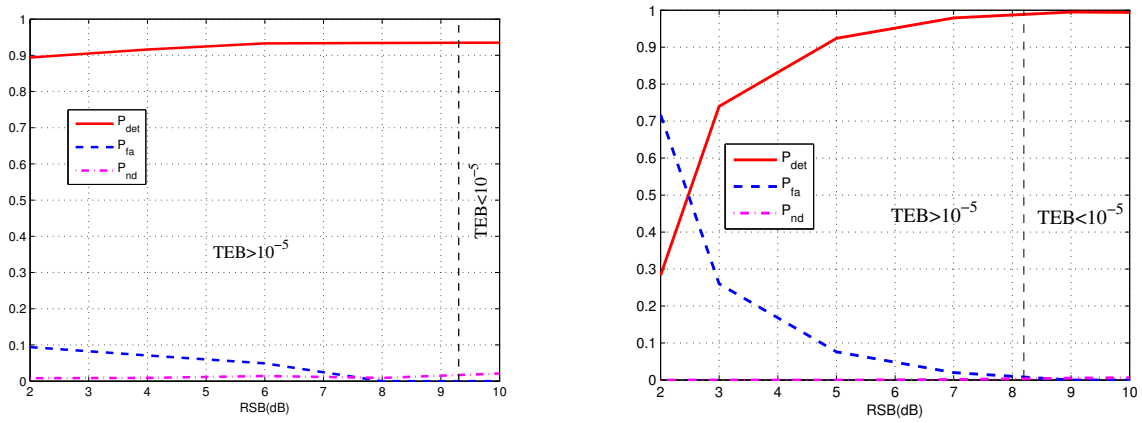
(a) LDPC(6,3) dans GF(2)

(b) LDPC(12,6) dans GF(2)

Figure 4.21 — TEB des codes LDPC(6,3) et LDPC(12,6) dans GF(2)

Sur la figure 4.22, les trois probabilités (P_{det}, P_{fa} et P_{nd}) en fonction du RSB(dB) sont représentées pour les deux codes LDPC(6,3) dans GF(2) et LDPC(12,6) dans GF(2). Nous avons aussi délimité sur chaque figure les zones où le TEB est supérieur/inférieur à 10^{-5}. Nous remarquons que pour les deux codes notre algorithme itératif offre d'excellentes performances. En effet, pour les zones correspondant à un TEB inférieur à 10^{-5}, la probabilité de détection est proche de 1.

Nous remarquons, pour les deux codes, que les probabilités de non-détection sont quasiment nulles même dans la zone où le TEB est supérieur à 10^{-5}. Nous constatons que les probabilités de fausse alarme pour le code LDPC(6,3) sont faibles et ne dépassent pas 0.1 dans la zone de TEB$>10^{-5}$ et nulles dans celle où TEB$<10^{-5}$. Pour le code LDPC(12,6), nous observons qu'à partir d'un certain seuil, ces probabilités deviennent supérieures aux probabilités de détection. En comparant le point de croisement de ces deux courbes aux TEB, nous notons que ces croisements se réalisent lorsque lorsque le TEB est supérieur à 10^{-2}. Par conséquent, ce code ne sera pas utilisé en pratique dans ce contexte (TEB$>10^{-2}$).



(a) LDPC(6,3) dans GF(2)

(b) LDPC(12,6) dans GF(2)

Figure 4.22 — P_{det}, P_{fa} et P_{nd} pour l'algorithme itératif d'identification

4.4.4.2 Gain apporté par notre algorithme itératif

Dans le but de limiter le temps de calcul permettant d'atteindre une bonne probabilité de détection, le nombre d'itérations maximum est fixé à 8. Afin d'évaluer ce nombre d'itérations, nous

noterons $\lambda_{x\to y}$, le gain obtenu entre l'itération x et y, tel que :

$$\lambda_{x\to y} = \frac{P_{det}(y) - P_{det}(x)}{P_{det}(x)} \tag{4.85}$$

où $P_{det}(i)$ est la probabilité de détecter les bonnes $n-k$ équations de parité à la i-ème itération. Ce gain sera exprimé en pourcentage.

Pour le code LDPC(6,3) dans GF(2), nous avons représenté sur la figure 4.23 la probabilité de détection P_{det} en fonction du RSB(dB) pour les itérations 1, 3 et 5.

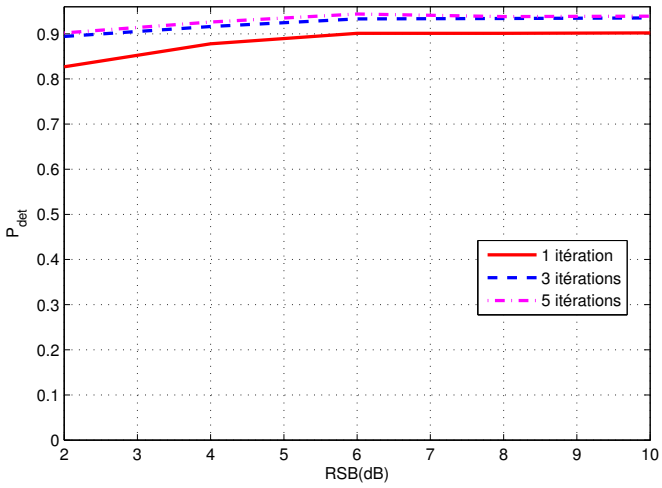


Figure 4.23 — P_{det} du nouvel algorithme itératif pour le code LDPC(6,3) pour $it = 1, 3, 5$

Nous pouvons remarquer que les performances optimales de notre algorithme sont atteintes à la 3-ème itération. En effet, le gain entre l'itération 3 et 5 est proche de 0. Cependant, le gain indiqué dans le tableau 4.9 entre l'itération 1 et 5 est compris entre 4 et 9%. Pour RSB(dB)=2 dB, le gain $\lambda_{1\to 5}$ est proche de 9%. Pour ce RSB(dB), après une itération P_{det} est proche de 0.82 et nous passons à 0.90 après 5 itérations.

RSB(dB)	2	4	10
$\lambda_{1\to 3}$ (%)	8	4.3	3.6
$\lambda_{1\to 5}$ (%)	9	5.5	4

Tableau 4.9 — Gain de détection pour le code LDPC(6,3)

Pour le code LDPC(12,6) dans GF(2), nous avons représenté sur la figure 4.24 la probabilité de détection P_{det} en fonction du RSB(dB) pour les itérations 1, 5 et 8.

Pour ce code, les performances optimales de notre algorithme sont atteintes à la 5-ème itération. Nous remarquons que pour RSB(dB)$<$ 4 le gain apporté par notre algorithme itératif est très important. Nous représentons sur le tableau 4.10 un récapitulatif des différents gains.

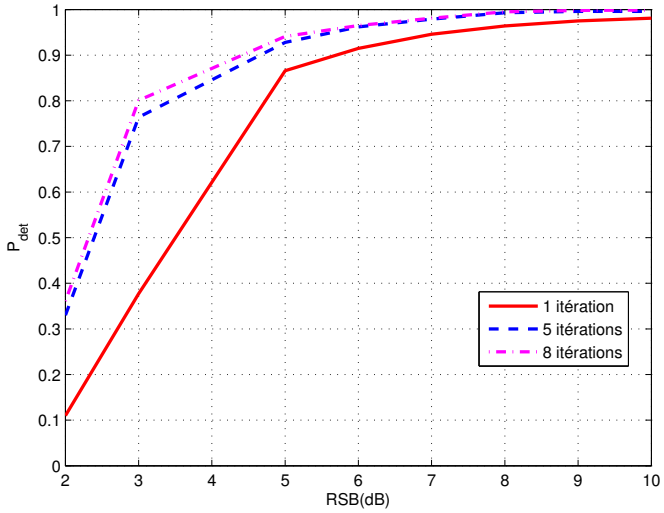


Figure 4.24 — P_{det} du nouvel algorithme itératif pour le code LDPC(12,6) pour $it = 1, 5, 8$

RSB(dB)	2	3	8
$\lambda_{1\to5}$ (%)	200	102	3
$\lambda_{1\to8}$ (%)	228	112	3

Tableau 4.10 — Gain de détection pour le code LDPC(12,6)

4.5 Conclusion

Dans ce chapitre, nous avons tout d'abord présenté trois méthodes d'identification aveugle de la taille des mots de code pour des codes linéaires binaires et non-binaires. La première méthode est une généralisation d'une méthode existante d'identification pour les codes binaires. Cette méthode est basée sur la recherche des colonnes presque dépendantes dans des matrices construites à partir des symboles reçus. Les deux autres méthodes sont basées sur l'analyse des comportements de la moyenne arithmétique et de la variance du nombre de 0 dans les matrices construites en utilisant l'élimination de Gauss dans GF(2^m).

Nous avons ensuite analysé les performances de nos trois méthodes. Nous avons montré qu'elles ont la même complexité. En revanche, les méthodes de la moyenne et de la variance sont plus robustes que la première méthode puisqu'elles n'ont pas besoin d'estimer la probabilité d'erreur du canal. En comparant les performances de détection, nous avons mis en avant l'influence du cardinal du corps de Galois $q = 2^m$ et l'augmentation de la taille des mots de code n sur nos méthodes d'identification. Nous avons montré que la méthode de la moyenne et de la variance offrent d'excellentes performances pour les codes LDPC testés dans GF(q), même en augmentant le cardinal du corps. Cependant, pour des tailles de code plus grandes, on a besoin de plus de données afin d'obtenir d'excellentes probabilités de détection. Le problème de la complexité de nos méthodes fait l'objet de nos travaux futurs. Pour différents types des canaux (canal q-aire symétrique, canal BBAG et canal de Rayleigh à trajets multiples) et différentes modulations (PAM et QAM), nous avons montré que la méthode de la variance offre les meilleurs performances de détection.

Pour identifier une base du code dual, nous avons supposé que les paramètres du code n et k et les paramètres du corps de Galois sont connus. Nous avons présenté une méthode d'identification pour les codes non-binaires en bloc basée sur l'utilisation d'un démodulateur à décision ferme. Cette

méthode est une généralisation d'une méthode d'identification pour les codes binaires.

Afin d'améliorer les performances de détection des $n-k$ équations de parité, nous avons tout d'abord présenté deux étapes d'amélioration de l'algorithme classique d'identification pour les codes binaires en utilisant un démodulateur à décision souple. La première étape d'amélioration consiste à trier les lignes de la matrice construite à partir des mots de code bruités suivant le critère de fiabilité des mots de code, à savoir que les mots de code les moins entachés d'erreurs sont placés dans les premières lignes. Cette étape peut être aussi appliquée à l'algorithme d'identification pour les codes non-binaires. Nous avons présenté une deuxième opération de tri qui consiste à trier les mots de code suivant les fiabilités des bits, à savoir que les mots de code qui ont un nombre minimal de bits erronés sont placés dans les premières lignes. Dans la deuxième étape d'amélioration, nous avons introduit un processus itératif dans l'algorithme d'identification qui est basé sur l'échange d'information entre l'algorithme classique d'identification et l'algorithme de décodage à décision souple. L'algorithme du décodage prend en entrée les équations de parité estimées et il fournit en sortie des informations souples sur les mots de code corrigés qui seront utilisées à leur tour pour l'identification. Dans le but d'améliorer les performances de détection des équations de parité, nous avons modifié l'algorithme de décodage afin de prendre en compte les probabilités de fiabilité des équations de parité estimées. Nous avons utilisé ces probabilités pour éliminer les équations les moins fiables et garder uniquement les $n-k$ équations les plus plus fiables.

Nous avons ensuite analysé les performances de détection des algorithmes d'identification d'une base du code dual présentés. Nous avons montré que l'algorithme conjoint d'identification et de décodage à décision souple offre d'excellentes performances comparé à l'algorithme classique d'identification. Puis, nous avons mis en avant l'influence des itérations sur les performances de détection. Afin de montrer la pertinence de nos résultats, nous avons pris en compte le taux d'erreur binaire acceptable par les standards actuels qui doit être inférieur à 10^{-5}. Nous avons montré que dans ces conditions la probabilité de détection de notre nouvel algorithme est proche de 1.

Deuxième partie

Etude des fonctions de mapping-demapping

CHAPITRE 5

Formalisme mathématique des fonctions de mapping-demapping

5.1 Introduction

La théorie des corps finis [Men93] est actuellement présente dans de nombreux domaines de recherche, tels que les mathématiques mais également le traitement du signal, la cryptologie, l'informatique et les communications numériques. La théorie des corps finis est également présente dans une part non-négligeable des nouvelles technologies de l'information et des dispositifs ou logiciels de communications actuels. Une partie des algorithmes, des logiciels et des architectures informatiques embarqués dans ces applications sont basés sur des calculs dans des corps finis et plus précisément dans les extensions du corps GF(2) (le cas binaire). Il s'agit généralement d'utiliser toutes les bonnes propriétés de l'algèbre linéaire qui sont judicieusement exploitées pour résoudre des problèmes.

Par exemple, dans les applications de cryptologie ou liées aux codes correcteurs d'erreurs [Pre92], nous sommes souvent amenés à résoudre un système d'équations linéaires, ou à calculer le déterminant d'une matrice composée d'éléments d'un corps fini. Dans le traitement du signal pour les communications numériques, il est fréquent d'effectuer les calculs de la dimension de l'espace du signal ou de son dual [BSH06, MGB11]. En général, toutes ces propriétés et leur calculs sont liés au calcul du déterminant dans le corps fini et à ses propriétés, dont la principale est la nullité ou non nullité du déterminant. Mais ce qui est radicalement différent entre une approche purement mathématique et une approche plus pragmatique pour la mise en oeuvre d'un algorithme de traitement du signal appliqué par exemple à un code correcteur d'erreurs est de savoir comment sont représentés les éléments du corps fini pour effectuer les calculs dans la pratique.

D'un point de vue mathématique, la seule chose importante est d'être capable de représenter formellement les différents éléments du corps fini et de faire les calculs. Toutes les différentes représentations sont équivalentes. Mais, pour mettre en oeuvre ces calculs sur un ordinateur ou sur un dispositif embarqué comme un FPGA ou un micro-contrôleur, il est nécessaire de préciser la manière dont les éléments sont codés [Rao81]. Ce processus est appelé "mapping" dans le traitement du signal et il consiste à associer à chaque élément une représentation spécifique. Par exemple, dans le cas d'une représentation classique on considère le code binaire naturel, avec les bits les plus significatifs à gauche. Mais il est possible de faire l'inverse et de réaliser le codage avec les bits les plus significatifs à droite, ce qui modifie les façons d'effectuer ou d'implémenter les calculs. Il en est généralement de même pour les transmissions numériques où les données binaires sont regroupées pour former des symboles qui sont ensuite convertis en des éléments d'une constellation d'ordre élevé comme les modulations d'amplitude en quadrature, QAM (Quadrature Amplitude Modulation), ou les modulations par déplacement de phase, PSK (Phase-shift keying). En particulier, il existe

de nombreuses études dans le domaine de l'égalisation [Sat75, Jab92, Gu05] et des codes correcteurs d'erreurs non-binaires [RS, Rya91, DM98a] destinées spécialement à l'optimisation conjointe du schéma de codage et du mapping des modulations. Dans ce cas, il est nécessaire de déterminer le mapping optimal, par rapport à un critère donné, ce qui ne conduit pas toujours au mapping classique de Gray. En fait, le résultat obtenu dépend des codes et des décodeurs effectivement mis en oeuvre dans le cadre de l'application considérée [CCW+07].

En général, il existe une composition de deux ou plusieurs étapes successives de mapping (codage de symbole + mapping de constellation + ...) qui peuvent être vues comme une opération globale de mapping. Pour récupérer l'information codée, il est nécessaire d'effectuer l'opération complémentaire, appelée "demapping". La fonction qui correspond à la composition des deux opérations de mapping et de demapping (que nous désignerons fonction de "mapping-demapping") doit être égale à l'opérateur identité, noté id. Si tout va bien, il est alors possible de traiter les données reçues ou stockées. En revanche, que se passe-t-il si l'opération de demapping utilisée ne correspond pas à l'opération inverse de mapping ? Dans un tel cas, la fonction de mapping-demapping n'est pas égale à l'opérateur l'identité. Cela nous a amenés à nous interroger sur l'impact des différentes fonctions de mapping-demapping sur les calculs d'algèbre linéaire et les propriétés dans le corps de Galois.

La transmission du signal sur le canal mais également les problèmes de synchronisation [VA05, Hos10] peuvent produire involontairement ces phénomènes qui peuvent provoquer des rotations de constellations des symboles reçus ou une inversion de polarité sur la voie en phase et en quadrature et bien d'autres dysfonctionnements. Ils peuvent aussi être liés à des domaines spécifiques d'applications tels que la surveillance de spectre (spectrum sensing) ou l'interception des communications. Dans tels contextes, le récepteur ne connaît pas le mapping utilisé à l'émission [KCA11]. C'est aussi un problème qui peut se produire dans le domaine de la cryptographie et/ou la cryptanalyse, où on peut choisir, pour des raisons de sécurité, le mode de représentation des éléments et des polynômes utilisés pour générer tous les éléments du corps [DA06, GKPP06, SK10]. Ce choix conditionne la sélection des méthodes pour mettre en oeuvre les calculs, ce qui conduit à des modifications au niveau de l'implémentation, en particulier s'ils sont réalisés à partir de tableaux codés en dur.

Notre objectif est d'introduire un formalisme mathématique spécifique pour étudier l'impact de la fonction de mapping-demapping sur les calculs d'algèbre linéaire en termes de conservation de la dimension des espaces vectoriels et de manipulation des propriétés des éléments du corps de Galois, dans le cas des codes correcteurs d'erreurs non-binaires pour les communications numériques. En effet, les codes correcteurs d'erreurs introduisent de la redondance dans le train de données, ce qui conduit à des combinaisons linéaires entre les éléments de chaque mot de code. L'ensemble des mots de code définissent l'espace du code, dont la dimension est égale à la taille des mots d'information. Pour corriger et décoder correctement les mots de code et récupérer les données, il est nécessaire que la fonction de mapping-demapping ne modifie ni les propriétés du code, ni l'espace du code. Dans un tel contexte, la connaissance de l'influence de la fonction de mapping-demapping sur les propriétés de l'espace du code peut donner quelques informations et un critère important pour détecter les défauts ou les modifications durant le processus de transmission, ce qui permet de les corriger de la bonne manière.

Ce chapitre est organisé comme suit. La section 5.2 présente quelques notations et définitions mathématiques. Ensuite, les fonctions de mapping-demapping indépendantes de l'espace vectoriel sont étudiées dans la section 5.3. Dans la section 5.4, les fonctions de mapping-demapping dépendantes de l'espace vectoriel sont définies. Les fonctions particulières qui correspondent au changement du polynôme primitif et l'inversion des poids des éléments de $\mathrm{GF}(p^m)$ font l'objet de la section 5.5. Dans la section 5.6, l'existence de fonctions de mapping-démapping qui conservent les équations de parité de poids 2 est étudiée dans le cas des applications sur les codes correcteurs d'erreurs non-binaires. La généralisation au cas des relations de parité de poids supérieur à 2 est

également discutée.

5.2 Définitions et notations de base

D'un point de vue théorique, toutes les représentations de $\mathrm{GF}(p^m)$ sont isomorphes en tant que corps. Mais d'un point de vue concret, il existe beaucoup de possibilités d'implémentations d'un corps de Galois. Dans les applications, le corps de Galois s'écrit généralement sous forme du quotient de l'anneau de polynômes par un polynôme irréductible de degré m, noté $\mathbb{Z}/p\mathbb{Z}[X]$. Pour un corps de Galois donné, il existe généralement plusieurs polynômes irréductibles, dont certains sont plus commode à utiliser que d'autres. Les polynômes choisis le plus souvent sont les polynômes primitifs grâce à la propriété importante de leurs racines. Ainsi, toute racine d'un polynôme primitif engendre tous les éléments non-nuls du corps, et est appelée élément primitif. En pratique, chaque élément du corps de Galois s'expriment en un symbole interne par une façon *ad hoc* pour construire les listes de données qui peuvent être manipulées par un dispositif approprié. Il s'agit de l'étape de mapping. Ensuite, les objets qui en résultent doivent être décodés en des listes d'éléments du même corps. C'est l'étape du demapping.

Dans ce chapitre, nous évaluons l'effet d'une opération de demapping inappropriée sur les calculs linéaires. Le mauvais usage du demapping peut être vu, d'un point du vu formel, comme une bijection sur le corps de Galois où chaque élément de ce corps sera remplacé par son image au travers de la bijection. Nous noterons $d_\star$ l'effet sur des objets de données d'une bijection d sur un corps de Galois et $d_\star(E)$ son résultat sur un objet de données E.

Exemple 5.36.

Si $\mathbf{E}$ est une matrice construite à partir des coefficients dans $\mathrm{GF}(p^m)$, notés $e_{i,j}$, la matrice obtenue après le demapping par d est :

$$d_\star(\mathbf{E}) = (d(e_{i,j}))_{i,j}$$

5.3 Sous-groupe de mapping-demapping indépendant de l'espace vectoriel

Dans cette section, nous étudions l'existence des fonctions de mapping-demapping qui préservent la dimension de tout espace vectoriel.

5.3.1 Fonctions remarquables

Nous présentons quelques fonctions remarquables qui peuvent préserver la dimension d'un espace vectoriel : le morphisme de Frobenius, les applications de multiplication par un scalaire non nul de $\mathrm{GF}(p^m)$ et plus généralement le sous-groupe engendré par ces deux types de $\mathrm{GF}(p)$-applications linéaires bijectives.

Morphisme de Frobenius

Le morphisme de Frobenius σ est un mapping-demapping remarquable du corps de Galois $\mathrm{GF}(p^m)$ sur lui-même. Il envoie chaque x de $\mathrm{GF}(p^m)$ sur $\sigma(x) = x^p$ où p est la caractéristique du corps de Galois. Ce mapping-demapping préserve les lois $+$ et $\cdot$ du corps de Galois $\mathrm{GF}(p^r)$. De ce fait, il est un morphisme du corps. Par la suite, nous présentons l'impact du morphisme de Frobenius sur les calculs d'algèbre linéaire.

Théorème 5.10. *Soit* $\mathbf{M} = (m_{i,j})_{i,j}$ *une matrice carrée à coefficients dans GF(*p^m*). Le déterminant de* $\mathbf{M}$ *est préservé par le morphisme de Frobenius :*

$$\det(\sigma_\star(\mathbf{M})) = \det((\sigma(m_{i,j}))_{i,j}) = \sigma(\det((m_{i,j})_{i,j})) = \sigma(\det(\mathbf{M})) \tag{5.1}$$

où $\sigma_\star(\mathbf{M})$ *désigne la matrice carée* $(\sigma(m_{i,j}))_{i,j}$.

Démonstration. Dans la première étape de cette démonstration, nous montrons que le morphisme de Frobenius est une application bijective. Alors, il suffit tout d'abord de montrer qu'il est injectif. En effet, puisque $\sigma(1) = 1$, ce mapping préserve chaque élément du groupe cyclique additif engendré par 1 (le sous-corps $\mathrm{GF}(p)$). De ce fait, le morphisme de Frobenius est aussi une application $\mathrm{GF}(p)$ linéaire sur le corps de Galois $\mathrm{GF}(p^m)$. Cette application est considérée comme un espace vectoriel sur le corps $\mathrm{GF}(p)$. Mais l'inexistence de diviseur de 0 dans un corps implique que le noyau de l'application linéaire σ est réduit à zéro. Par conséquent, le mapping de Frobenius est injectif.

La dimension de cet espace vectoriel est m qui est fini. Alors le mapping de Frobenius est bijectif, ce qui implique que le morphisme de Frobenius peut constituer une possible application de mapping-demapping.

Étant donné que le morphisme de Frobenius σ est compatible avec l'arithmétique du corps de Galois, il envoie une expression algébrique sur la même expression, où chaque élément est remplacé par son image au travers du mapping-demapping de Frobenius. En particulier, l'image par σ du déterminant d'une matrice carrée $\mathbf{M} = (m_{i,j})_{i,j}$ à coefficients dans le corps de Galois $\mathrm{GF}(p^m)$ est donnée par l'équation (5.1). □

Ainsi, le morphisme de Frobenius conserve la nullité du déterminant donc la dimension des espaces vectoriels. Pour préciser l'impact du morphisme de Frobenius sur l'algèbre linéaire, considérons un entier naturel n et un sous espace vectoriel $\mathcal{C}$ sur $\mathrm{GF}(p^m)^n$.

Théorème 5.11. *L'image de l'opération de mapping-demapping induite par le morphisme de Frobenius* σ *de l'espace vectoriel* $\mathcal{C}$ *est un sous-espace vectoriel sur GF(*p^m*) qui est défini par :*

$$\sigma_\star(\mathcal{C}) = \left\{ (\sigma(x_i))_{1\leq i\leq n} \middle| (x_i)_{1\leq i\leq n} \in \mathcal{C} \right\} \tag{5.2}$$

Démonstration. Si $\mathbf{x} = (x_i)_{1\leq i\leq n}$ et $\mathbf{y} = (y_i)_{1\leq i\leq n}$ sont deux vecteurs de $\mathcal{C}$ et λ est un élément du corps de Galois $\mathrm{GF}(p^m)$, $\sigma(\mathbf{x}) + \lambda \cdot \sigma(\mathbf{y})$ est donné par :

$$\sigma(\mathbf{x}) + \lambda \cdot \sigma(\mathbf{y}) = \sigma(\mathbf{x} + \sigma^{-1}(\lambda) \cdot \mathbf{y}) \in \sigma(\mathcal{C})$$

Ainsi, $\sigma_\star(\mathcal{C})$ est un sous espace vectoriel de $\mathrm{GF}(p^m)^n$. □

Grâce à la bijectivité de σ, la dimension de $\mathcal{C}$ et $\sigma(\mathcal{C})$ est la même. On retrouve ainsi directement que le mapping-demapping de Frobenius σ préserve la dimension de tout espace vectoriel.

Multiplication par un élément non-nul

Considérons comme fonction de mapping-demapping, la multiplication par un élément non-nul du corps $\mathrm{GF}(p^m)$, *i.e.* l'automorphisme linéaire dans $\mathrm{GF}(p^m)$ sur l'espace vectoriel sur $\mathrm{GF}(p^m)$. Soit λ un élément non-nul du corps de Galois $\mathrm{GF}(p^m)$ et m_λ une application de $\mathrm{GF}(p^m)$ sur lui même qui envoie un élément x sur $m_\lambda(x) = \lambda \cdot x$.

Théorème 5.12. *La multiplication* m_λ *en tant que mapping-demapping préserve la dimension de tout espace vectoriel et l'espace vectoriel lui-même.*

Démonstration. Étant donnée que λ est inversible, le mapping-demapping m_λ est bijectif et le mapping-demapping de son inverse s'écrit $m_{1/\lambda}$. Grâce à la linéarité de $\mathrm{GF}(p^m)$, la multiplication par λ modifie chaque expression polynomiale homogène de degré k par le facteur λ^k. Par exemple, le calcul du déterminant en utilisant m_λ comme fonction de mapping-demapping de la matrice carrée $\mathbf{M} = (m_{i,j})_{i,j}$ de taille $(n \times n)$ à coefficients dans le corps $\mathrm{GF}(p^m)$ conduit à :

$$\det(m_{\lambda\star}(\mathbf{M})) = \lambda^n \cdot \det \mathbf{M}$$

où $m_{\lambda\star}(\mathbf{M})$ désigne la matrice carrée à coefficients $(m_\lambda(m_{i,j}))_{i,j}$. On peut en déduire :

$$\det(m_{\lambda\star}(\mathbf{M})) = 0 \Longleftrightarrow \det \mathbf{M} = 0$$

Par conséquent, le mapping-demapping m_λ préserve la dimension de tout espace vectoriel.

Montrons maintenant que m_λ préserve l'espace vectoriel lui-même. En effet, considérons $\mathcal{C}$ un espace vectoriel sur le corps $\mathrm{GF}(p^m)$. Puisque λ est inversible, l'image de $\mathcal{C}$ par la fonction de mapping-demapping m_λ est :

$$\begin{aligned}
m_{\lambda\star}(\mathcal{C}) &= \left\{(m_\lambda(x_i))_{1\leq i\leq n} \middle| (x_i)_{1\leq i\leq n} \in \mathcal{C}\right\} \\
&= \left\{(\lambda \cdot (x_i))_{1\leq i\leq n} \middle| (x_i)_{1\leq i\leq n} \in \mathcal{C}\right\} \\
&= \left\{(y_i)_{1\leq i\leq n} \middle| \tfrac{1}{\lambda}(y_i)_{1\leq i\leq n} \in \mathcal{C}\right\} \\
&= \left\{(y_i)_{1\leq i\leq n} \middle| (y_i)_{1\leq i\leq n} \in \mathcal{C}\right\} \\
&= \mathcal{C}.
\end{aligned}$$

De ce fait, l'espace vectoriel $\mathcal{C}$ est préservé par la fonction m_λ. □

Groupe généré par le morphisme de Frobenius et l'automorphisme de multiplication

Dans les paragraphes précédents, deux applications de mapping-demapping ont été présentées : le morphisme de Frobenius σ et la multiplication par un élément non-nul m_λ. Ces deux types d'applications de mapping-demapping préservent la dimension de tout espace vectoriel puisqu'elles conservent la nullité et la non-nullité du determinant. Ces deux propriétés sont conservées par la composition et l'inversion des applications de mapping-demapping pour lesquelles ces propriétés sont vérifiées. Ainsi, tout élément du groupe d'applications de mapping-demapping engendré par σ et m_λ préserve la dimension de tout espace vectoriel. Ces applications de mapping-demapping seront appelées *fonctions de mapping-demapping universelles*. Soit $\overline{\mathbb{MD}}$ le plus petit groupe pour la loi de composition contenant à la fois le morphisme de Frobenius et les automorphismes de multiplication. Cette partie se concentre sur l'étude des propriétés de ce groupe.

Théorème 5.13. *Le groupe $\overline{\mathbb{MD}}$ possède les propriétés suivantes :*

- *Tout élément de $\overline{\mathbb{MD}}$ admet une et une seule écriture sous la forme :*

$$\overline{\mathbb{MD}} = \left\{ m_\lambda \circ \sigma^k \;\middle|\; \lambda \in GF(p^m)^* \text{ et } k \in [\![0, m[\![\right\} \tag{5.3}$$

- *La fonction réciproque d'un élément g de ce groupe s'écrit :*

$$g^{-1} = m_{\frac{1}{\sigma^{-k}(\lambda)}} \circ \sigma^{-k} \tag{5.4}$$

- *L'ordre (cardinal) de ce groupe est donné par :*

$$|\overline{\mathbb{MD}}| = m \cdot (p^m - 1) \tag{5.5}$$

Démonstration. Nous démontrons que toute fonction de mapping-demapping universelle g s'écrit sous $m_\lambda \circ \sigma^k$, avec $\lambda \in \mathrm{GF}(p^m)^*$ et $k \in [\![0, m[\![$. En effet, grâce à l'additivité de générateurs de $\overline{\mathbb{MD}}$, g est une fonction additive, *i.e.* pour tout x et y éléments du corps de Galois $\mathrm{GF}(p^m)$, $g(x+y) = g(x) + g(y)$. En outre, le morphisme de Frobenius dans $\mathrm{GF}(p^m)$ est d'ordre m pour la loi de composition :

$$\underbrace{\sigma \circ \sigma \circ \cdots \circ \sigma}_{m} = \sigma^m = id$$

De plus, pour tous les éléments non-nuls λ et λ' de $\mathrm{GF}(p^m)$:

$$\begin{cases} m_\lambda^{-1} = m_{\lambda^{-1}} \\ m_\lambda \circ m_{\lambda'} = m_{\lambda \cdot \lambda'} \end{cases}$$

Ces deux propriétés traduisent le fait que l'ensemble de fonctions m_λ correspond à un groupe multiplicatif $(GF(p^m)^*, \cdot)$ par l'isomorphisme qui envoie m_λ sur λ. Ainsi, pour tout élément non-nul λ et tout élément x de $\mathrm{GF}(p^m)$, $\sigma(m_\lambda(x))$ est donné par :

$$\sigma(m_\lambda(x)) = \sigma(\lambda \cdot x) = \lambda^p \cdot x^p = m_{\sigma(\lambda)}(\sigma(x)).$$

Nous pouvons déduire que $\sigma \circ m_\lambda$ and $m_\lambda \circ \sigma$ peuvent s'écrire sous la forme :

$$\begin{cases} \sigma \circ m_\lambda = m_{\sigma(\lambda)} \circ \sigma \\ m_\lambda \circ \sigma = \sigma \circ m_{\sigma^{-1}(\lambda)} \end{cases}$$

Par conséquent, le groupe universel $\overline{\mathbb{MD}}$ peut être défini par : $\overline{\mathbb{MD}} = \Big\{ m_\lambda \circ \sigma^k \;\Big|\; \lambda \in GF(p^m)^* \text{ and } k \in [\![0, m[\![\Big\}$.

La fonction réciproque de $g = m_\lambda \circ \sigma^k$ est donnée par :

$$\begin{aligned} g^{-1} &= (m_\lambda \circ \sigma^k)^{-1} \\ &= \sigma^{-k} \circ m_{\frac{1}{\lambda}} \\ &= m_{\frac{1}{\sigma^{-k}(\lambda)}} \circ \sigma^{-k} \end{aligned}$$

Pour déterminer le cardinal de $\overline{\mathbb{MD}}$, il est nécessaire de démontrer que toute fonction de mapping-demapping universelle peut s'écrire d'une façon unique sous la forme $m_\lambda \circ \sigma^k$. Si nous considérons $m_\lambda \circ \sigma^k = id$, alors nous aurons :

$$m_\lambda = \sigma^{-k} \Longrightarrow m_\lambda(1) = \sigma^{-k}(1) \Longrightarrow \lambda = 1$$

Dans ce cas, $m_\lambda \circ \sigma^k = id$ permet d'obtenir :

$$\begin{cases} \lambda = 1 \\ \sigma^k = id \end{cases} \Longleftrightarrow \begin{cases} \lambda = 1 \\ k = 0 \end{cases}$$

Donc, l'expression $m_\lambda \circ \sigma^k$ est unique. Pour cela, il existe exactement $|\overline{\mathbb{MD}}| = m \cdot (p^m - 1)$ applications universelles dans $\mathrm{GF}(p^m)$. □

Puisque les fonctions de mapping-demapping qui préservent la dimension de tout espace vectoriel forment un groupe et tout groupe est défini par une loi de composition, alors la caractérisation

abstraite de la loi de composition du groupe universel $\overline{\mathbb{MD}}$ est souhaitable. Elle est donnée par le théorème 5.14.

Théorème 5.14. *Soit α un élément primitif du corps de Galois GF(p^m). Considérons l'application qui envoie chaque couple (i,k), avec i dans $[\![0,p^m-1[\![$ et k dans $[\![0,m[\![$, sur $m_{\alpha^i}\circ\sigma^k$. Cette application est un isomorphisme entre le produit semi-direct de $\mathbb{Z}/(p^m-1)\rtimes\mathbb{Z}/m\mathbb{Z}$ et le groupe des fonctions de mapping-demapping universelles $\overline{\mathbb{MD}}$ tel que la loi du produit semi-direct est définie par :*

$$(i,k)(i',k') = (i+i'\cdot p^k, k+k') \tag{5.6}$$

Démonstration. La loi du groupe est tout simplement la contrepartie de l'égalité suivante :

$$\begin{aligned}(m_{\alpha^i}\circ\sigma^k)\circ(m_{\alpha^{i'}}\circ\sigma^{k'}) &= m_{\alpha^i}\circ m_{\sigma^k(\alpha^{i'})}\circ\sigma^k\circ\sigma^{k'}\\ &= m_{\alpha^i}\circ m_{(\alpha^{i'})^{p^k}}\circ\sigma^{k+k'}\\ &= m_{\alpha^i\cdot\alpha^{i'\cdot p^k}}\circ\sigma^{k+k'}\\ &= m_{\alpha^{i+i'\cdot p^k}}\circ\sigma^{k+k'}.\end{aligned}$$

□

5.3.2 Groupe conservant la dimension de tout espace vectoriel

Nous avons mentionné précédemment qu'il existe des fonctions de mapping-demapping comme le morphisme de Frobenius, la multiplication par des éléments non nuls et le groupe engendré par ces deux types de fonctions qui préservent la dimension de tout espace vectoriel. Le but de cette section est de déterminer toutes les fonctions de mapping-démapping qui préservent la dimension de tout espace vectoriel dans GF(p^m).

Théorème 5.15. *L'ensemble de fonctions de mapping-demapping qui préserve la dimension de tout espace vectoriel est le groupe universel de fonctions de mapping-demapping $\overline{\mathbb{MD}}$.*

Démonstration. Soit d une fonction de mapping-demapping, *i.e.* une application bijective du corps GF(p^m) sur lui même. Nous cherchons une fonction de mapping-demapping d telle que pour tout espace vectoriel V sur GF(p^m) :

$$\dim d_\star(V) = \dim V$$

où $d_\star(V)$ définit l'espace vectoriel généré par $d(\mathbf{x})$ avec $\mathbf{x}$ dans V.

La dimension d'un espace vectoriel peut être calculée en utilisant les déterminants des matrices carrées construites par des vecteurs de cet espace. La dimension correspond à la taille de la plus grande matrice de déterminant non-nul. Alors, la conservation de la dimension d'un espace vectoriel par un mapping-demapping d est équivalente à la propriété suivante :

Propriété 5.6. *Pour toute matrice carrée $\mathbf{M} = (m_{i,j})_{i,j}$,*

$$\det(\mathbf{M}) = 0 \Longleftrightarrow \det(d_\star(\mathbf{M})) = 0$$

où $d_\star(\mathbf{M}) = (d(m_{i,j}))_{i,j}$.

Si nous supposons que d préserve la dimension de tout espace vectoriel, les déterminants de matrices de tailles (1×1), (2×2) et (3×3) seront aussi préservés. En fait, le choix de ces trois matrices est suffisant pour déduire la propriété de conservation de la dimension par le mapping-demapping d. Pour cette raison, il n'est pas utile de prendre des matrices de taille plus grande.

- La conservation de la nullité du déterminant de la matrice $(\mathbf{0})$ conduit à $d(\mathbf{0}) = \det(d_\star(\mathbf{0})) = 0$.

- Considérons x et y deux éléments du corps de Galois $\mathrm{GF}(p^m)$. Alors le déterminant de la matrice ci-dessous de taille (2×2) est donné par :

$$\det \begin{pmatrix} 1 & y \\ x & x \cdot y \end{pmatrix} = 0 \Longrightarrow \det \begin{pmatrix} d(1) & d(y) \\ d(x) & d(x \cdot y) \end{pmatrix} = 0 \tag{5.7}$$

  Ainsi, l'équation (5.7) permet de déduire que $d(1) \cdot d(x \cdot y) = d(x) \cdot d(y)$. Comme la fonction d est bijective et $d(0) = 0$, alors $d(1) \neq 0$. Soit $\tilde{d}$ l'application du corps de Galois $\mathrm{GF}(p^m)$ sur lui même, tel que pour tout x de $\mathrm{GF}(p^m)$:

$$\tilde{d}(x) = \frac{d(x)}{d(1)} \tag{5.8}$$

  Nous démontrons que cette application préserve la dimension de tout espace vectoriel. On a $\tilde{d}(x) = m_{1/d(1)} \circ d$ donc $\tilde{d}$ est la composition de deux bijections. Elle est donc bijective.

  La composition de deux fonctions de mappings-demapping conserve la dimension de tout espace vectoriel. Donc, $\tilde{d}$ conserve la dimension de tout espace vectoriel.

  Puisque $\tilde{d}(1) = 1$, pour tous éléments x et y de $\mathrm{GF}(p^m)$, $\tilde{d}(x \cdot y) = \tilde{d}(x) \cdot \tilde{d}(y)$. L'application $\tilde{d}$ est donc un automorphisme du groupe multiplicatif $(GF(p^m)^*, \cdot)$.

- Soit x et y deux éléments de $\mathrm{GF}(p^m)$. Comme $\tilde{d}(0) = 0$ et $\tilde{d}(1) = 1$, le déterminant de la matrice ci-dessous de taille (3×3) est donné par :

$$\det \begin{pmatrix} x & 1 & 0 \\ y & 0 & 1 \\ x+y & 1 & 1 \end{pmatrix} = 0 \Longrightarrow \det \begin{pmatrix} \tilde{d}(x) & 1 & 0 \\ \tilde{d}(y) & 0 & 1 \\ \tilde{d}(x+y) & 1 & 1 \end{pmatrix} = 0 \tag{5.9}$$

  Cette équation permet de déduire que $\tilde{d}(x+y) = \tilde{d}(x) + \tilde{d}(y)$. Alors, l'application $\tilde{d}$ est un automorphisme du groupe additif $(GF(p^m), +)$. Donc, cette application est un automorphisme du corps, *i.e.* un élément du groupe de Galois de $\mathrm{GF}(p^m)$ qui est généré par le morphisme de Frobenius σ. Donc, on peut prendre un entier k tel que $\tilde{d} = \sigma^k$.

Comme $d = m_{d(1)} \circ \tilde{d} = m_{d(1)} \circ \sigma^k$, le mapping-demapping d qui conserve la dimension de tout espace vectoriel est un élément du groupe universel de fonctions de mapping-demapping $\overline{\mathbb{MD}}$. □

5.3.3 Exemple de fonctions de mapping-demapping universelles dans $\mathrm{GF}(2^2)$

Dans le domaine des communications numériques, on utilise le corps de Galois de cardinal 2^m. Dans cette partie, nous prenons l'exemple du corps de Galois $\mathrm{GF}(2^2)$. Soit α une racine du polynôme primitif $P(X) = X^2 + X + 1$, *i.e.* $\alpha^2 = \alpha + 1$. Cette racine est un élément primitif de $\mathrm{GF}(2^2) = \{0, 1, \alpha, \alpha^2\}$.

En utilisant l'équation (5.5), le groupe des fonctions de mapping-demapping universelles possède $2 \cdot (2^2 - 1) = 6$ éléments représentés dans le tableau 5.1.

Tableau 5.1 — Fonctions universelles dans $\mathrm{GF}(2^2)$

$id = m_1$:	0	$\mapsto$	0	σ :	0	$\mapsto$	0
	1	$\mapsto$	1		1	$\mapsto$	1
	α	$\mapsto$	α		α	$\mapsto$	α^2
	α^2	$\mapsto$	α^2		α^2	$\mapsto$	α
m_α :	0	$\mapsto$	0	$m_\alpha \circ \sigma$:	0	$\mapsto$	0
	1	$\mapsto$	α		1	$\mapsto$	α
	α	$\mapsto$	α^2		α	$\mapsto$	1
	α^2	$\mapsto$	1		α^2	$\mapsto$	α^2
m_{α^2} :	0	$\mapsto$	0	$m_{\alpha^2} \circ \sigma$:	0	$\mapsto$	0
	1	$\mapsto$	α^2		1	$\mapsto$	α^2
	α	$\mapsto$	1		α	$\mapsto$	α
	α^2	$\mapsto$	α		α^2	$\mapsto$	1

Pour le corps de Galois $\mathrm{GF}(2^2)$, $\overline{\mathbb{MD}}$ contient exactement toutes les bijections qui conservent 0. Dans ce petit corps, cette condition est équivalente à la linéarité sur $\mathrm{GF}(2) = \{0, 1\}$.

5.4 Fonctions de mapping-demapping conservant la dimension de certains espaces vectoriels

Les fonctions de mapping-demapping universelles préservent dans toutes les conditions la dimension des espaces vectoriels. Nous avons montré qu'elles sont les seules qui puissent le faire. Mais les translations sont également remarquables de ce point de vue. En effet, celles-ci ne modifient pas beaucoup les dimensions : elles conservent la dimension de certains espaces vectoriels et ajoutent 1 pour les autres. Cette section est consacrée à l'étude de ces fonctions particulières.

5.4.1 Exemple de fonctions de mapping-demapping non-universelles : les translations

Dans cette partie, nous étudions le cas des fonctions de mapping-demapping remarquables qui parfois ajoutent 1 à la dimension d'un espace vectoriel et parfois conservent cette dimension. Nous allons établir les résultats pour une classe de fonctions apparues lors de nos simulations.

Théorème 5.16. *Soit μ un élément non-nul du corps GF(p^m) et τ_μ la translation par μ sur GF(p^m) :*

$$x \mapsto x + \mu \tag{5.10}$$

Soit V un sous-espace vectoriel de GF(p^m)^n sur GF(p^m). Soit $W =< \tau_{\mu\star}(V) >$ le petit espace vectoriel sur GF(p^m) qui contient tous $(\tau_\mu(x_i))_i$, où $(x_i)_i$ est dans V.

$$\dim W = \begin{cases} \dim V & si\ (\mathbf{1})_i \in V \\ \dim V + 1 & si\ (\mathbf{1})_i \notin V \end{cases} \tag{5.11}$$

où $(\mathbf{1})_i$ est un vecteur ayant la même taille que $(x_i)_i$.

Démonstration. Soit $\mathbf{x} = (x_i)_i$ un vecteur de V. En appliquant le mapping défini dans l'équation (5.10) à ce vecteur, nous avons :

$$(\tau_\mu(x_i))_i = (x_i + \mu)_i = (x_i)_i + (\mu)_i = (x_i)_i + \mu \cdot (\mathbf{1})_i$$

Les vecteurs $(\mathbf{1})_i$ et $(x_i)_i$ appartiennent à W car ils peuvent s'écrire en fonction de τ_μ :

$$\begin{cases} (\mathbf{1})_i = \frac{1}{\mu}(\tau_\mu(\mathbf{0}))_i \\ (x_i)_i = (\tau_\mu(x_i))_i - (\tau_\mu(\mathbf{0}))_i \end{cases}$$

Ainsi, l'espace vectoriel W peut être défini en fonction de V par : $W = V + GF(p^m)(\mathbf{1})_i$.

Par conséquent, la dimension de W est égale à la dimension de V si $(\mathbf{1})_i$ est un vecteur de V. Sinon elle est égale à la dimension de V plus 1. □

5.4.2 Sous-groupe de fonctions de mapping-demapping quasi-universelles

Au cours de nos simulations, un ensemble plus large de fonctions de mapping-demapping remarquables est apparu. Il préserve la dimension de certains espaces vectoriels et il ajoute 1 aux autres dimensions. Cet ensemble sera appelé groupe des fonctions de mapping-demapping quasi-universelles et noté $\widetilde{\mathbb{MD}}$. Il est composé des translations et leur compositions avec les fonctions de mapping-demapping universelles.

Définition 5.19. *Une fonction de mapping-demapping d est dite quasi-universelle s'il existe une fonction de mapping-demapping universelle $m_\lambda \circ \sigma^k$ et un élément μ du corps $GF(p^m)$ tel que pour tout x dans $GF(p^m)$:*

$$d(x) = m_\lambda \circ \sigma^k(x) + \mu$$

Dans ce cas, d peut être exprimé par :

$$d = \tau_\mu \circ m_\lambda \circ \sigma^k$$

où τ_μ est la translation par μ.

D'après la définition 5.19, les fonctions de mapping-demapping quasi-universelles sont les translations de fonctions universelles par des constantes.

Théorème 5.17. *Le nombre de fonctions de mapping-demapping quasi-universelles est donné par :*

$$|\widetilde{\mathbb{MD}}| = m \cdot p^m \cdot (p^m - 1) \tag{5.12}$$

L'équation (5.12) est démontrée ci-dessous.

Démonstration. L'écriture de la fonction quasi-universelle d sous la forme $d = \tau_\mu \circ m_\lambda \circ \sigma^k$ est unique, grâce à $\mu = d(0)$ et à l'unicité de l'écriture de la fonction universelle $\tau_{-\mu} \circ d = m_\lambda \circ \sigma^k$ de $\overline{\mathbb{MD}}$. Le nombre de fonctions de mapping-demapping quasi-universelles est $|\overline{\mathbb{MD}}| \times |GF(p^m)| = m \cdot p^m \cdot (p^m - 1)$. □

Parmi $m \cdot p^m \cdot (p^m - 1)$ fonctions de mapping-demapping quasi-universelles, il existe exactement $m \cdot (p^m - 1)$ fonctions universelles (d'après l'équation (5.5)). Ainsi, il y a exactement $m \cdot (p^m - 1)^2$ fonctions quasi-universelles mais non-universelles. Ces fonctions seront appelées des fonctions strictement quasi-universelles.

Théorème 5.18. *L'ensemble des fonctions de mapping-demapping quasi-universelles est stable par l'inversion et la composition. Il forme donc un groupe. L'application qui envoie chaque couple*

(μ, g) du GF$(p^m) \times \overline{\mathbb{MD}}$ sur $\tau_\mu \circ g$ est un isomorphisme entre le produit semi-direct de GF$(p^m) \rtimes \overline{\mathbb{MD}}$ et le groupe de fonctions de mapping-demapping quasi-universelles $\widetilde{\mathbb{MD}}$ où la loi du produit semi-direct est définie par :

$$(\mu, g)(\mu', g') = (\mu + g(\mu'), g \circ g') \tag{5.13}$$

La fonction réciproque $d = \tau_\mu \circ g$, notée d^{-1}, avec $g = m_\lambda \circ \sigma^k$, est donnée par :

$$d^{-1} = \tau_{-g^{-1}(\mu)} \circ g^{-1} \tag{5.14}$$

avec $g^{-1} = m_{\frac{1}{\sigma^{-k}(\lambda)}} \circ \sigma^{-k}$.

Démonstration. Soit μ un élément de GF(p^m) et f une fonction de $\overline{\mathbb{MD}}$. Considérons x un élément du GF(p^m). De par l'additivité de toute fonction de mapping-demapping universelle, $(g \circ \tau_\mu)(x)$ peut s'écrire :

$$(g \circ \tau_\mu)(x) = g(x + \mu) = g(x) + g(\mu) = (\tau_{g(\mu)} \circ g)(x)$$

Ainsi, $g \circ \tau_\mu$ peut aussi s'écrire comme $\tau_{g(\mu)} \circ g$. L'inverse de $(\tau_\mu \circ g = \tau_\mu \circ m_\lambda \circ \sigma^k)$ est donnée par :

$$\begin{aligned} d^{-1} &= (\tau_\mu \circ g)^{-1} \\ &= g^{-1} \circ \tau_{-\mu} \\ &= \tau_{-g^{-1}(\mu)} \circ g^{-1} \end{aligned}$$

où g^{-1} est défini dans l'équation (5.4).

La composition de deux fonctions quasi-universelles $\tau_\mu \circ g$ et $\tau_{\mu'} \circ g'$, où g et g' sont universelles, peut se réécrire sous :

$$\begin{aligned} (\tau_\mu \circ g) \circ (\tau_{\mu'} \circ g') &= \tau_\mu \circ \tau_{g(\mu')} \circ g \circ g' \\ &= \tau_{\mu + g(\mu')} \circ (g \circ g'). \end{aligned}$$

D'où, le groupe des fonctions de mapping-demapping quasi-universelles $\widetilde{\mathbb{MD}}$ est défini par la loi du produit semi-direct donnée par : $(\mu, g)(\mu', g') = (\mu + g(\mu'), g \circ g')$. □

Théorème 5.19. *Soit $\tau_\mu \circ g$ une fonction de mapping-demapping strictement quasi-universelle. Soit V un sous espace vectoriel de GF$(p^m)^n$ sur GF(p^m). La dimension du plus petit espace vectoriel sur GF(p^m) contenant tous $((\tau_\mu \circ g)(x_i))_i$ où $(x_i)_i$ dans V est la dimension de V si $(\mathbf{1})_i$ appartient à $g_\star(V)$. Sinon celle-ci est égale à la dimension de V plus 1.*

Démonstration. Nous démontrons que les deux conditions : $(\mathbf{1})_i \in g_\star(V)$ et $(\mathbf{1})_i \in V$, sont équivalentes. Soit $g = m_\lambda \circ \sigma^k$. Si $(\mathbf{1})_i \in V$ alors :

$$(\mathbf{1})_i = \frac{1}{\lambda}(\lambda)_i = \frac{1}{\lambda}(\lambda \cdot 1^{p^k})_i = \frac{1}{\lambda}(g(1))_i \in g_\star(V)$$

D'où, $(\mathbf{1})_i \in V \Longrightarrow (\mathbf{1})_i \in g_\star(V)$. Cette implication s'écrivant avec g^{-1} à la place de g et $g_\star(V)$ à la place de V mène à :

$$(\mathbf{1})_i \in g_\star(V) \Longrightarrow (\mathbf{1})_i \in g_\star^{-1}(g_\star(V)) = V$$

Ainsi, nous obtenons l'équivalence : $(\mathbf{1})_i \in g_\star(V) \Longleftrightarrow (\mathbf{1})_i \in V$. Par conséquent, d'après le théorème 5.16, la dimension du plus petit sous-espace vectoriel sur GF(p^m) contenant tous $((\tau_\mu \circ g)(x_i))_i$ où $((x_i))_i$ dans V est la dimension de V si $(\mathbf{1})_i$ appartient à V. Sinon celle-ci est égale à la dimension de V plus 1. □

5.4.3 Exemple d'application de fonctions quasi-universelles dans GF(2^m)

L'ordre du groupe formé par toutes les fonctions de mapping-demapping dans le corps GF(p^m), noté ici $\mathbb{M}\mathrm{D}$, c'est-à-dire le groupe symétrique sur GF(p^m), est $p^m!$. Les ordres du groupe universel $\overline{\mathbb{M}\mathrm{D}}$, du groupe quasi-universel $\widetilde{\mathbb{M}\mathrm{D}}$ et du groupe de toutes les fonctions de mapping-demapping $\mathbb{M}\mathrm{D}$ pour les corps GF(2^2), GF(2^3) et GF(2^4) sont présentés dans le tableau 5.2.

Tableau 5.2 — Ordres de $\overline{\mathbb{M}\mathrm{D}}$ et $\widetilde{\mathbb{M}\mathrm{D}}$ en fonction de GF(2^m)

GF(2^m)	$\lvert\overline{\mathbb{M}\mathrm{D}}\rvert$	$\lvert\widetilde{\mathbb{M}\mathrm{D}}\rvert$	$\lvert\mathbb{M}\mathrm{D}\rvert$
GF(2^2)	6	24	24
GF(2^3)	21	168	40320
GF(2^4)	60	960	$16! \simeq 2.1 \cdot 10^{13}$

La proportion d'applications quasi-universelles parmi les fonctions de mapping-demapping décroît significativement avec le cardinal du corps de Galois. Elle est donnée par :

$$\frac{m}{(p^m - 2)!}$$

Il s'agit d'une application sur 240 dans le cas du corps GF(2^3) et seulement d'une application sur environ $2.1795 \cdot 10^{10}$ dans le cas du corps GF(2^4).

Prenons l'exemple du corps GF(2^2). Le groupe des fonctions de mapping-demapping quasi-universelles possède exactement 24 éléments. Dans ce cas, il s'agit du nombre total de fonctions de mapping-demapping ($24 = 4!$) dans ce petit corps de Galois GF(2^2). Donc, toute application bijective sur GF(2^2) est quasi-universelle. L'ensemble de fonctions strictement quasi-universelles de mapping-demapping possède $24 - 6 = 18$ éléments. Par la suite, nous noterons ces fonctions par $g + \mu$ au lieu de $\tau_\mu \circ g$, où $g = m_\lambda \circ \sigma^k$. Quelques exemples de fonctions de mapping-demapping strictement quasi-universelles sont présentés dans le tableau 5.3.

Tableau 5.3 — Quelques fonctions strictement quasi-universelles dans GF(2^2)

$$id + 1 : \begin{array}{ccc} 0 & \mapsto & 1 \\ 1 & \mapsto & 0 \\ \alpha & \mapsto & \alpha^2 \\ \alpha^2 & \mapsto & \alpha \end{array} \qquad id + \alpha : \begin{array}{ccc} 0 & \mapsto & \alpha \\ 1 & \mapsto & \alpha^2 \\ \alpha & \mapsto & 0 \\ \alpha^2 & \mapsto & 1 \end{array}$$

$$id + \alpha^2 : \begin{array}{ccc} 0 & \mapsto & \alpha^2 \\ 1 & \mapsto & \alpha \\ \alpha & \mapsto & 1 \\ \alpha^2 & \mapsto & 0 \end{array} \qquad m_\alpha \circ \sigma + 1 : \begin{array}{ccc} 0 & \mapsto & 1 \\ 1 & \mapsto & \alpha^2 \\ \alpha & \mapsto & 0 \\ \alpha^2 & \mapsto & \alpha \end{array}$$

$$m_\alpha \circ \sigma + \alpha : \begin{array}{ccc} 0 & \mapsto & \alpha \\ 1 & \mapsto & 0 \\ \alpha & \mapsto & \alpha^2 \\ \alpha^2 & \mapsto & 1 \end{array}$$

5.5 Fonctions de mapping-demapping particulières

Dans cette section, nous étudions deux fonctions de mapping-demapping particulières qui peuvent modifier la représentation des symboles du corps de Galois et peuvent modifier également le résultat des calculs dans l'algèbre linéaire. Ce sont les fonctions de mapping-demapping qui changent le polynôme primitif et les fonctions qui permutent par symétrie les coordonnées de la représentation des éléments du corps de Galois.

5.5.1 Changement du polynôme primitif

Parmi les processus erronés pour les symboles d'un corps de Galois, le changement du polynôme primitif a une certaine importance. L'utilisation d'un mauvais polynôme primitif perturbe profondément les calculs dans l'algèbre linéaire.

Théorème 5.20. *Le changement du polynôme primitif n'est jamais une fonction de mapping-demapping quasi-universelle.*

Démonstration. Puisque le groupe quasi-universel comprend les fonctions universelles et les fonctions strictement quasi-universelles, nous démontrons tout d'abord que le changement du polynôme primitif n'est pas une fonction de mapping-demapping universelle.

Soit α un élément primitif du corps $\mathrm{GF}(p^m)$, i.e. un générateur du groupe cyclique $(GF(p^m)^*, \cdot)$ des éléments non-nuls de ce corps fini. Une fonction de mapping-demapping qui correspond au changement d'élément primitif envoie α sur un autre élément primitif β, i.e. $\beta = \alpha^i$ où i est premier par rapport à $(p^m - 1)$. Les fonctions de mapping-demapping correspondant au changement d'élément primitif sont exactement les automorphismes sur le groupe multiplicatif $(GF(p^m)^*, \cdot)$ étendus en envoyant 0 sur 0.

Soit i un entier de $[\![1, p^m - 1[\![$ premier par rapport à $p^m - 1$ et d la fonction de mapping-demapping correspondant au changement d'élément primitif α en un élément primitif $\beta = \alpha^i$. Cette fonction est telle que :

$$d : \begin{cases} 0 \mapsto 0, \\ \\ \alpha^j \mapsto \beta^j = \alpha^{i \cdot j} \quad \forall j \in [\![0, p^m - 1[\![. \end{cases} \tag{5.15}$$

Nous appliquons la démonstration par l'absurde. Nous supposons que la fonction de mapping-demapping d appartient au groupe universel de fonctions de mapping-demapping $\overline{\mathbb{MD}}$. Soit λ un élément de $\mathrm{GF}(p^m)^*$ et k un élément de $[\![0, m[\![$ tel que $d = m_\lambda \circ \sigma^k$. L'élément 1 peut être obtenu par :

$$1 = \beta^0 = d(\alpha^0) = d(1) = m_\lambda(\sigma^k(1)) = \lambda \cdot 1^{(p^k)} = \lambda$$

De ce fait, le mapping-demapping d est égal à σ^k puisque $\lambda = 1$. Mais, σ est un automorphisme du corps :

$$x \text{ est une racine de } P \iff \sigma^k(x) \text{ est une racine de } P$$

Cette équivalence permet de déduire que $\beta = \sigma^k(\alpha)$ est une racine du polynôme minimal de α qui est le polynôme primitif choisi. Par conséquent, la fonction de mapping-demapping d ne change pas le polynôme primitif. Désormais, le changement du polynôme primitif ne peut pas être universel. En revanche, est-il strictement quasi-universel ?

Comme mentionné dans la section 5.4.2, les fonctions de mapping-demapping strictement quasi-universelles ne préservent pas le zéro du determinant. Elles envoient 0 sur μ, où μ est un élément

non-nul de GF(p^m). Puisque la fonction de mapping-demapping qui correspond au changement du polynôme primitif préserve le zéro, *i.e* $0 \mapsto 0$, elle ne peut pas être strictement quasi-universelle. Par conséquent, elle n'est pas une fonction quasi-universelle. □

En particulier, si le polynôme primitif P n'est pas symétrique *i.e.* $X^m P(1/X) \neq P(X)$, la fonction de mapping-demapping qui change l'ordre des coefficients de P n'est pas quasi-universelle.

Par exemple, le corps de Galois GF(2^3) comporte exactement deux polynômes primitifs de degré 3 : $P_1(X) = X^3 + X + 1$ et $P_2(X) = X^3 + X^2 + 1$. La représentation binaire de ces deux polynômes est donnée par : $P_1 = \begin{pmatrix}1 & 0 & 1 & 1\end{pmatrix}$ et $P_2 = \begin{pmatrix}1 & 1 & 0 & 1\end{pmatrix}$. Nous pouvons voir d'après cette représentation que les polynômes P_1 et P_2 ne sont pas symétriques. Alors, l'expression de $P_2(X)$ peut être obtenue par celle de $P_1(X)$:

$$P_2(X) = X^3 + X^2 + 1 = X^3 \cdot \left(1 + \frac{1}{X} + \frac{1}{X^3}\right) = X^3 \cdot P_1\left(\frac{1}{X}\right) \tag{5.16}$$

Soit α une racine du polynôme primitif $P_2(X)$, *i.e.* $\alpha^3+\alpha^2+1 = 0$, et β une racine de $P_1(X)$. D'après l'équation (5.16), $1/\alpha = 1 \cdot \alpha^{-1} = \alpha^7 \cdot \alpha^{-1} = \alpha^6$ est une racine de $P_1(x)$, *i.e.* $\alpha^{-3} + \alpha^{-1} + 1 = 0$, puisque $\alpha \neq 0$. De ce fait, β est égal à α^6. Nous donnons dans le tableau 5.4 les différentes représentations de symboles obtenus par la fonction de mapping-demapping qui correspond au changement de polynôme primitif $P_2 \to P_1$, notée d, et définie par l'équation (5.15), avec $\beta = \alpha^6$.

Tableau 5.4 — Impact du changement de polynôme primitif dans GF(2^3)

Représentations des symboles générés par P_2			Représentations des symboles convertis et reconvertis par d		
Puissance de α	Binaire	Entier	Puissance de β	Binaire	Entier
0	(0 0 0)	0	0	(0 0 0)	0
1	(0 0 1)	1	1	(0 0 1)	1
α	(0 1 0)	2	$\beta = \alpha^6$	(1 1 0)	6
α^2	(1 0 0)	4	$\beta^2 = \alpha^5$	(0 1 1)	3
α^3	(1 0 1)	5	$\beta^3 = \alpha^4$	(1 1 1)	7
α^4	(1 1 1)	7	$\beta^4 = \alpha^3$	(1 0 1)	5
α^5	(0 1 1)	3	$\beta^5 = \alpha^2$	(1 0 0)	4
α^6	(1 1 0)	6	$\beta^6 = \alpha$	(0 1 0)	2

5.5.2 Inversion des poids dans la représentation des symboles du corps de Galois

Dans cette partie, nous déterminons les fonctions de mapping-demapping qui peuvent permuter les coordonnées de la représentation des éléments du corps GF(p^m) par symétrie lorsqu'ils sont représentés en utilisant la base de $(\alpha^i)_{0 \leq i < m}$, où α est un élément primitif. Cette opération de permutation correspond à l'inversion des poids de la représentation des symboles du corps de Galois. Chaque coordonnée de cette représentation est pondérée par une puissance de α. Un tel élément de GF(p^m) est représenté comme une somme de ses coordonnées multipliés par la puissance correspondante de α. Dans la représentation classique, la coordonnée la moins significative est normalement à droite et celui la plus significative est à gauche. Ces positions peuvent être modifiées par la fonction étudiée ici, ce qui perturbe les calculs dans l'algèbre linéaire. La question qui se pose est de savoir si la fonction de mapping-demapping qui permute les coordonnées de la représentation de la base $(\alpha^i)_{0 \leq i < m}$ appartient ou non au groupe quasi-universel $\widetilde{\mathbb{M}\mathrm{D}}$. Le théorème ci-dessous peut répondre à cette question.

Théorème 5.21. *Soit α un élément primitif de GF(p^m) et d une fonction qui réalise la permutation des coordonnés par symétrie dans la base $(\alpha^i)_{0\leq i<m}$, une application de GF(p^m) sur lui-même, définie par :*

$$\sum_{0\leq i<r} a_i \cdot \alpha^i \mapsto \sum_{0\leq i<r} a_{r-1-i} \cdot \alpha^i$$

où a_i est un élément du GF(p).

La fonction d appartient au sous-groupe universel dans le cas de $p = 2$ et $m = 2$, sinon elle n'est pas quasi-universelle.

Démonstration. Nous supposons que d est une fonction quasi-universelle. Alors, elle peut s'écrire sous la forme : $d = \tau_\mu \circ m_\lambda \circ \sigma^k$, où λ est un élément non-nul de GF(p^m), k est un entier dans $[\![0, m[\![$ et μ est un élément non-nul de GF(p^m).

- Puisque d préserve 0, *i.e.* $d(0) = 0$, alors $\mu = 0$. Ainsi, d peut s'écrire : $d = m_\lambda \circ \sigma^k$. Cette expression permet de déduire que d est universelle.
- On a $d(1) = \alpha^{m-1}$ et $m_\lambda\left(\sigma^k(1)\right) = 1$, alors $\lambda = \alpha^{m-1}$.
- Comme d est universelle, elle préserve le déterminant. Ainsi, puisque :

  $$\det\begin{pmatrix} 1 & \alpha \\ \frac{1}{\alpha} & 1 \end{pmatrix} = 0$$

  alors :

  $$\det\begin{pmatrix} d(1) & d(\alpha) \\ d\left(\frac{1}{\alpha}\right) & d(1) \end{pmatrix} = 0$$

  ce qui implique que :

  $$\det\begin{pmatrix} \alpha^{m-1} & \alpha^{m-2} \\ d\left(\frac{1}{\alpha}\right) & \alpha^{m-1} \end{pmatrix} = 0 \Rightarrow \alpha^{2\cdot m-2} = \alpha^{m-2} \cdot d\left(\frac{1}{\alpha}\right)$$

  De ce fait, on a $d\left(\frac{1}{\alpha}\right) = \alpha^m$.
- D'après $d\left(\frac{1}{\alpha}\right) = \alpha^m$, on peut déduire :

  $$\begin{aligned} d\left(\tfrac{1}{\alpha}\right) &= \alpha^m \quad \Leftrightarrow \quad m_{\alpha^{m-1}}\left(\sigma^k\left(\tfrac{1}{\alpha}\right)\right) = \alpha^m \\ &\Leftrightarrow \quad \alpha^{m-1-p^k} = \alpha^m \\ &\Leftrightarrow \quad 1 = \alpha^{p^k+1} \end{aligned}$$

  Ainsi, $p^m - 1$ divise $p^k + 1$. Puisque $p^k + 1 > 0$, on a $p^k + 1 \geq p^m - 1 \Rightarrow p^k \geq p^m - 2$. Or, on a $k \leq m - 1$. Donc, $p^{m-1} \geq p^m - 2$, *i.e.* $1 \geq p - \dfrac{2}{p^{m-1}}$. Par conséquent, $p = 2$ et $m - 1 \leq 1$. Comme $m > 1$, on aura $m = 2$.

Mais, si $m = 2$, comme $d(0) = 0$, alors d est universelle. Sinon elle n'est pas quasi-universelle. □

5.6 Conservation des équations de parité de poids 2

Dans le cas d'applications sur les codes correcteurs d'erreurs non-binaires, nous essayons dans cette partie de déterminer les fonctions de mapping-demapping qui transforment une relation de parité donnée de poids 2 en une relation de même poids. L'objectif est d'illustrer, à travers un exemple simple, les contraintes qui pèsent sur les fonctions de mapping-demapping pour préserver la dimension de certains espaces vectoriels.

Soit $\mathcal{R}_\alpha$ une équation de parité sous la forme $Y = \alpha \cdot X$ dans un espace vectoriel de dimension finie sur GF(p^m). Soit d une fonction de mapping-demapping qui envoie toute hypersurface de vecteurs vérifiant $\mathcal{R}_\alpha$ sur une hypersurface définie par une relation de parité $\mathcal{R}_\beta : Y = \beta \cdot X$. Une telle fonction de mapping-demapping d est une bijection sur le corps GF(p^m) qui vérifie :

$$\forall x \in GF(p^m), \quad d(\alpha \cdot x) = \beta \cdot d(x) \tag{5.17}$$

Toutes les fonctions de mapping-demapping préservent les relations de parité sous la forme $Y = X$. Ainsi, il n'y a pas de limitation pour les fonctions de mapping-demapping dans le cas $\alpha = 1$. Désormais, supposons que α est un élément du corps GF(p^m) différent de 0 et 1. Par la suite, nous proposons un théorème qui révèle le nombre et le type de fonctions de mapping-demapping vérifiant (5.17).

Théorème 5.22. *Les fonctions de mapping-demapping qui préservent les équations de parité de poids* 2 *ont la forme :*

$$\begin{array}{ccl} GF(p^m) & \longrightarrow & GF(p^m) \\ 0 & \mapsto & 0, \\ x_i \cdot \alpha^k & \mapsto & y_i \cdot \beta^k \quad \forall i \in [\![1, s]\!], \forall k \in [\![0, t[\![. \end{array}$$

où $\{x_1, ..., x_s\}$ *et* $\{y_1, ..., y_s\}$ *sont deux systèmes d'éléments représentatifs des* s *classes modulo* $H = \{1, \alpha, \alpha^2, \cdots, \alpha^{t-1}\}$ *du groupe multiplicatif* $(GF(p^m)^*, \cdot)$*, où* t *est l'ordre de* α *et* $s = \frac{p^m-1}{t}$ *est le nombre de classes modulo le sous groupe cyclique généré par* α*. Le nombre de ces fonctions, noté* N_t*, est déterminé par :*

$$N_t = \varphi(t) \cdot t^s \cdot s! \tag{5.18}$$

où $\varphi(t)$ *est la fonction d'Euler.*

Démonstration. Soit d une fonction de mapping-demapping qui satisfait la propriété (5.17). Dans ce cas, l'élément 0 est préservé par d. En effet, le demapping de 0 par d est donné par :

$$d(0) = d(0 \cdot \alpha) = \beta \cdot d(0)$$

Ce résultat montre que $d(0)$ peut être égal à 0, *i.e.* $d(0) = 0$, ou $\beta = 1$. Mais, si on suppose que $\beta = 1$, nous aurons $d(\alpha) = d(1)$. Grâce à la bijectivité de d, cette hypothèse est exclue. Ainsi, l'élément 0 doit être préservé par d, *i.e.* $d(0) = 0$.

Nous démontrons maintenant que α et β ont le même ordre. Soit k un entier multiple de l'ordre de α, *i.e.* $\alpha^k = 1$. Alors, $d(1)$ peut être exprimé par :

$$d(1) = d(\alpha^k) = \beta^k \cdot d(1)$$

Or, on a $d(1) \neq d(0) = 0$. Alors, β^k doit être égal à 1. Soit l un entier multiple de l'ordre de β, *i.e.* $\beta^l = 1$. Alors $d(\alpha^l)$ peut être exprimé par :

$$d(\alpha^l) = \beta^l \cdot d(1) = d(1)$$

La bijectivité de d permet d'avoir $\alpha^l = 1$. Donc, α et β ont le même ordre dans le groupe multiplicatif des éléments non-nuls du corps de Galois. Soient t cet ordre et $s = \frac{p^m-1}{t}$ le nombre de classes modulo le sous-groupe H généré par α, *i.e.* $H = \{1, \alpha, \alpha^2, \cdots, \alpha^{t-1}\}$.

Grâce à la cyclicité du groupe multiplicatif $(GF(p^m)^*, \cdot)$, H est l'unique groupe de cardinal t. De fait, H est généré aussi par β, *i.e.* $H = \{1, \beta, \beta^2, \cdots, \beta^{t-1}\}$.

Soit $\{\mathbf{x}_1, ..., \mathbf{x}_s\}$ et $\{\mathbf{y}_1, ..., \mathbf{y}_s\}$ deux systèmes des éléments représentatifs de s classes modulo H du groupe multiplicatif $GF(p^m)^*$, *i.e.* $\mathbf{x}_i = \{x_i, x_i \cdot \alpha, x_i \cdot \alpha^2, \cdots, x_i \cdot \alpha^{t-1}\}$ et $\mathbf{y}_i = \{y_i, y_i \cdot \beta, y_i \cdot$

$\beta^2, \cdots, y_i \cdot \beta^{t-1}\}$ pour $i \in [\![1, s]\!]$. Soit f la fonction bijective définie par :

$$\begin{array}{rccl} f: & GF(p^m) & \longrightarrow & GF(p^m) \\ & 0 & \mapsto & 0, \\ & x_i \cdot \alpha^k & \mapsto & y_i \cdot \beta^k \quad \forall i \in [\![1, s]\!], \forall k \in [\![0, t[\![. \end{array}$$

Vérifions si la fonction f satisfait la propriété (5.17) même pour des éléments non-nuls. Pour $x_i = 0$, son image par f est donnée par :

$$f(0 \cdot \alpha) = f(0) = 0 = \beta \cdot f(0)$$

Ainsi, la condition (5.17) est satisfaite par f pour l'élément 0. Dans le cas d'un élément non-nul, $x_i \in GF(p^m)^*$, nous considérons les entiers $i \in [\![1, s]\!]$ et $k \in [\![0, t[\![$. L'image de $\alpha \cdot x_i \cdot \alpha^k$ par f est donnée par :

$$\begin{aligned} f(\alpha \cdot x_i \cdot \alpha^k) &= f(x_i \cdot \alpha^{k+1}) = \begin{cases} y_i \cdot \beta^{k+1} & \text{si } k \neq t-1, \\ y_i & \text{si } k = t-1 \end{cases} \\ &= \beta^{k+1} \cdot y_i = \beta \cdot f(x_i \cdot \alpha^k). \end{aligned}$$

Donc, la fonction de mapping-demapping f satisfait la condition (5.17). Comme mentionné précédemment, les éléments α et β doivent avoir le même ordre. Alors, les fonctions de mapping-demapping recherchées qui préservent les équations de parité de poids 2 représentent exactement l'ensemble de toutes les fonctions f. Nous déterminons, maintenant, le nombre de ces fonctions.

A l'aide des analyses précédentes, une fonction de mapping-demapping f, qui satisfait la condition (5.17) pour un β approprié, est définie par β et $\{y_1, ..., y_s\}$ où l'élément α et l'ensemble $\{x_1, ..., x_s\}$ sont préalablement définis. Les différents choix de β et $\{y_1, ..., y_s\}$ permettent d'obtenir différentes fonctions f.

Le choix de β est limité aux éléments de même ordre t que α. De ce fait, le nombre possible de β est la fonction d'Euler de t, $\varphi(t)$, où t est son ordre. Le nombre de $\{y_1, ..., y_s\}$ peut être calculé de manière itérative par l'ajout d'un élément y_{i+1} à l'ensemble $\{y_1, ..., y_i\}$. L'élément y_{i+1} doit être choisi en dehors des $i \cdot t$ éléments de i classes y_1, ..., y_i modulo H dans GF$(p^m)^*$. D'où, le nombre de systèmes représentatifs de classes modulo H, $\{y_1, ..., y_s\}$, est :

$$\prod_{0 \leq i < s} \left(p^m - 1 - i \cdot t\right) = \prod_{0 \leq i < s} (s \cdot t - i \cdot t) = t^s \cdot s!$$

Par conséquent, le nombre de fonctions de mapping-demapping qui satisfont la condition (5.17) pour un β approprié est $N_t = \varphi(t) \cdot t^s \cdot s!$ tel que $t \cdot s = p^m - 1$. □

Puisque les fonctions de mapping-demapping qui sont étudiées dans cette section préservent l'élément 0, nous pouvons déduire que les fonctions universelles en font partie.

Le tableau 5.5 présente quelques exemples de nombre de fonctions possibles de mapping-demapping qui conservent les équations de parité de poids 2 en fonction de t dans les cas de GF(2^2), GF(2^3) et GF(2^4). Ce nombre est calculé en utilisant l'équation (5.18).

D'après ce tableau, nous pouvons observer que toutes les fonctions qui conservent les relations de parité du poids 2 dans le cas du corps GF(2^2) sont les fonctions de mapping-demapping universelles. Étant donné le faible nombre de fonctions de mapping-demapping qui sont capables de conserver une relation de parité de poids 2, cela suggère qu'il y a encore moins de fonctions pour vérifier les relations de poids supérieur ou égal à 3. Nous pouvons constater que la borne inférieure du nombre des fonctions étudiées ici est évidemment le cardinal du sous-groupe universel $\overline{\mathbb{MD}}$. Nous pensons aussi que pour tout code ayant des équations de parité de poids supérieur à 2, certains problèmes ou erreurs qui conduisent à l'utilisation implicite d'une fonction de mapping-demapping différente

Tableau 5.5 — Nombre de fonctions de mapping-demapping conservant les équations de parité

$\mathrm{GF}(2^r)$	t	N_t
$\mathrm{GF}(2^2)$	3	$N_3 = 6$
$\mathrm{GF}(2^3)$	7	$N_7 = 42$
$\mathrm{GF}(2^4)$	3	$N_3 = 58320$
	5	$N_5 = 3000$
	15	$N_{15} = 120$

qui n'appartient pas au sous-groupe universel $\overline{\mathbb{MD}}$, peut être détecté car ces problèmes peuvent provoquer un changement de dimension du code dual et la dimension du code lui-même.

5.7 Conclusion

Dans ce chapitre, nous avons développé un formalisme mathématique afin d'étudier l'impact d'un mauvais demapping sur la chaîne de transmission et sur les calculs linéaires dans le corps de Galois. Nous avons présenté et étudié les caractéristiques de deux sous-ensembles de fonctions de mapping-demapping qui font partie d'un groupe appelé quasi-universel. Le premier sous-ensemble qui correspond à un groupe des fonctions de mapping-demapping universelles est caractérisé par la conservation de la dimension de tout espace vectoriel. Le deuxième sous-ensemble qui est strictement quasi-universel conserve la dimension de certains espaces vectoriel du code lorsque le mot $(\mathbf{1})_i$ composé seulement de 1 appartient au code, sinon il ajoute 1 à cette dimension. Nous avons étudié des cas particuliers de fonctions de mapping-demapping comme le changement de polynôme primitif et d'inversion du poids fort et du poids faible. Pour la fonction du changement du polynôme primitif, nous avons montré que cette fonction n'est pas quasi-universelle. De même pour la fonction de l'inversion des poids, nous a avons montré qu'elle n'est pas quasi-universelle sauf pour le corps $\mathrm{GF}(2^2)$.

Dans ce chapitre, nous avons traité aussi le cas des fonctions de mapping-demapping qui permettent de conserver les équations de parité de poids 2 afin de trouver leur nombre qui reste assez faible par rapport au nombre total de fonctions de mapping-demapping. Ce formalisme nous permet de déduire qu'on peut détecter quelques défauts de transmission et des représentations erronées par le calcul de la dimension de l'espace vectoriel engendré.

CHAPITRE 6

Impact des fonctions de mapping-demapping sur la détection de défauts et l'identification d'un code

6.1 Introduction

Dans ce chapitre, nous allons illustrer un intérêt de notre formalisme, présenté dans le chapitre précédent. Il nous permet de déduire des propriétés clés sur des effets des fonctions de mapping-demapping sur les calculs dans les corps de Galois en les appliquant à deux contextes différents. La première application est la détection de certains défauts de transmission au niveau d'un récepteur de signaux de communications numériques où les données sont protégées par un code correcteur d'erreurs non-binaire connu et utilisant des modulations numériques à grand nombre d'états de type QAM ou PSK. La seconde application porte sur la détection d'une mauvaise interprétation du codage utilisé au niveau des calculs effectués en regroupant les bits représentant les symboles d'un corps de Galois lors d'une application manipulant des données protégées par un code correcteur d'erreurs non-binaire connu. Nous allons montrer l'impact des fonctions du groupes quasi-universel et de certaines fonctions particulières non quasi-universelles sur l'identification en aveugle des codes correcteurs d'erreurs non-binaires dans un contexte d'une transmission non-bruitée.

6.2 Exemple de détection de défauts rencontrés au niveau du récepteur d'une transmission numérique

Dans cette section, nous considérons à titre illustratif une transmission numérique en utilisant une modulation QAM d'ordre 4 où les données sont protégées par un code correcteur d'erreurs non-binaire construit sur GF(2^2). Dans ce contexte, les symboles de GF(2^2)= $\{0, 1, \alpha, \alpha^2\}$, où α est une racine de polynôme primitif $x^2 + x + 1$, peuvent être directement associés aux symboles de la modulation à 4 états. La constellation de ce mapping est représentée sur la figure 6.1.

Étant donné que le code est connu, sa dimension est alors connue au niveau du récepteur et ne devrait pas changer lorsque le bon demapping est utilisé en réception et permet d'interpréter correctement les symboles reçus. Il est alors possible de recevoir plusieurs mots de code et de calculer la dimension engendrée par eux à travers le calcul du rang d'une matrice formée par ces mots de code. Nous allons maintenant regarder ce qui se passe si certains défauts courants, présentés ci-dessous, se produisent lors d'une transmission :

1. Inversion de polarité sur la voie en phase,

2. Inversion de polarité sur la voie en quadrature,
3. Inversion de polarité sur la voie en phase et en quadrature ou rotation de π des symboles de la constellation,
4. Rotation de $+\frac{\pi}{2}$ des symboles de la constellation,
5. Rotation de $-\frac{\pi}{2}$ des symboles de la constellation.

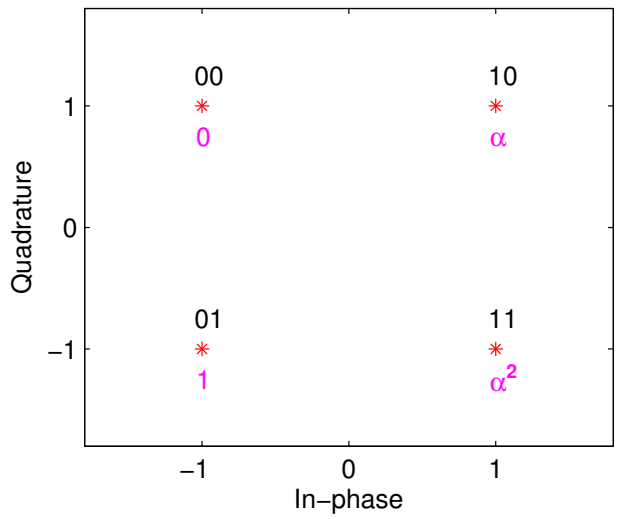


Figure 6.1 — Diagramme de la constellation 4-QAM choisie comme référence

Ces défauts sont illustrés sur la figure 6.2. Sur cette figure, les diagrammes de constellation des symboles de 4-QAM ainsi que leur trajets sont représentés. En particulier, les problèmes 3, 4 et 5 peuvent être engendrés par des défauts de synchronisation et de déphasage. Grâce à notre formalisme, nous sommes en mesure de déterminer les fonctions de mapping-demapping qui correspondent à ces défauts. Il s'agit des fonctions $id+\alpha$ pour le défaut 1, $id+1$ pour le défaut 2, $id+\alpha^2$ pour le défaut 3, $m_\alpha \circ \sigma + \alpha$ pour le défaut 4 et $m_\alpha \circ \sigma + 1$ pour le défaut 5 comme il est également possible de les voir sur la figure 6.2.

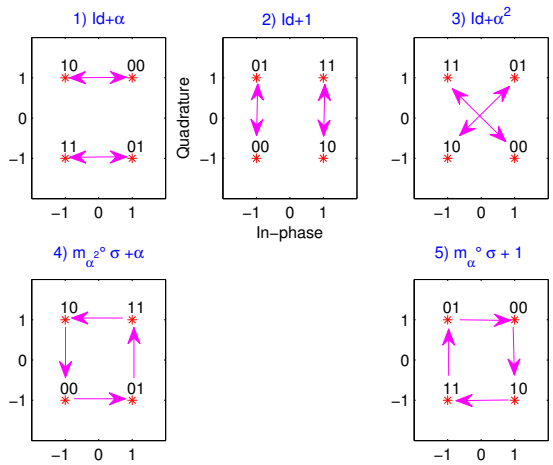


Figure 6.2 — Constellation de symboles 4-QAM pour des fonctions strictement quasi-universelles

Notre formalisme nous permet également, au regard des propriétés énoncées dans le chapitre précédent, de dire que toutes les fonctions de mapping-demapping qui correspondent à ces défauts courants de transmission appartiennent à l'ensemble des fonctions strictement quasi-universelles. Nous pouvons également affirmer d'après le théorème 5.12 que ces défauts peuvent être détectés à condition que le mot comportant uniquement des 1 ($(\mathbf{1})_i$) ne fait pas partie des mots de code, car sinon il ne sera pas possible de voir un changement de dimension avec une augmentation de 1. Afin de corriger ces défauts, il suffira d'appliquer les différentes fonctions de mapping-demapping réciproques définies par l'équation (5.14) et de regarder pour laquelle la dimension diminue de 1. La fonction réciproque de mapping-demapping qui fera chuter la dimension permet de corriger le défaut et décaler les symboles sur les points de la constellation utilisés à l'émission. La fonction réciproque appropriée est celle qui nous permet d'obtenir la bonne dimension du code. Mais, il est possible d'obtenir plus qu'une fonction de mapping-demapping ayant cette propriété. Elles font partie du sous-groupe universel. Dans ce cas, nous cherchons celle qui correspond à la fonction id.

Dans le tableau 6.1, nous avons représenté les différentes fonctions qui sont obtenues en appliquant toutes les fonctions réciproques d^{-1} aux fonctions de mapping-demapping correspondant aux défauts courants de transmission.

	Fonctions de mapping-demapping correspondant aux défauts courants d				
Fonctions d^{-1}	$id+1$	$id+\alpha$	$id+\alpha^2$	$m_\alpha \circ \sigma + 1$	$m_\alpha \circ \sigma + \alpha$
$id+1$	id	$id+\alpha^2$	$id+\alpha$	$m_\alpha \circ \sigma$	$m_\alpha \circ \sigma + \alpha^2$
$id+\alpha$	$id+\alpha^2$	id	$id+1$	$m_\alpha \circ \sigma + \alpha^2$	$m_\alpha \circ \sigma$
$id+\alpha^2$	$id+\alpha$	$id+1$	id	$m_\alpha \circ \sigma + \alpha$	$m_\alpha \circ \sigma + 1$
$m_\alpha \circ \sigma + 1$	$m_\alpha \circ \sigma$	$m_\alpha \circ \sigma + \alpha^2$	$m_\alpha \circ \sigma + \alpha$	id	$id+\alpha^2$
$m_\alpha \circ \sigma + \alpha$	$m_\alpha \circ \sigma + \alpha^2$	$m_\alpha \circ \sigma$	$m_\alpha \circ \sigma + 1$	$id+\alpha^2$	id

Tableau 6.1 — Fonctions obtenues par application des fonctions réciproques d^{-1}

Nous pouvons remarquer que les fonctions résultantes de mapping-demapping qui permettent d'obtenir la bonne dimension du code sont id ou $m_\alpha \circ \sigma$ (cellules grises). Les autres fonctions réciproques engendrant des fonctions de l'ensemble strictement quasi-universel ne conservent pas la dimension de tout code. Donc, elles peuvent être éliminées. Ainsi, la fonction appropriée est parmi celles qui permettent d'obtenir la bonne dimension du code. Nous pouvons constater que pour le défaut 3 qui correspond à la fonction $id+\alpha^2$, il existe une seule fonction résultante appartenant au sous-groupe universel qui est la fonction souhaitée id. La correction de ce défaut peut être réalisée par la fonction réciproque $id+\alpha^2$. Cependant, dans le cas des défauts 1, 2, 4 et 5, les fonctions résultantes qui permettent de conserver la dimension du code sont $m_\alpha \circ \sigma$ et id. Notre objective est maintenant de distinguer les deux fonctions résultantes afin d'identifier la fonction réciproque qui permet d'obtenir id. Notre idée consiste à introduire dans la trame de synchronisation une sous trame composée d'une succession de 1 puisque l'élément 1 n'est pas conservé par la fonction $m_\alpha \circ \sigma$ comme montré dans le tableau 5.1, $id(1) = 1$ et $m_\alpha \circ \sigma(1) = \alpha$. Nous appliquons les fonctions réciproques correspondant à $m_\alpha \circ \sigma$ et id à cette sous trame. La bonne fonction réciproque est celle qui conserve tous les éléments de la sous trame.

Il est également intéressant de voir si les fonctions de mapping-demapping du sous-groupe universel semblent être représentatives de certains défauts courants lors de la transmission car dans ce cas ces défauts ne peuvent jamais être détectés. La figure 6.3 nous montre les diagrammes de constellations de symboles obtenus par les fonctions de mapping-demapping universelles. Nous pouvons constater que hormis la fonction identité qui ne pose aucun problème, les autres fonctions ne peuvent pas être engendrées par des défauts courants de transmission dans le cas du corps

GF(2^2) et de modulation 4-QAM.

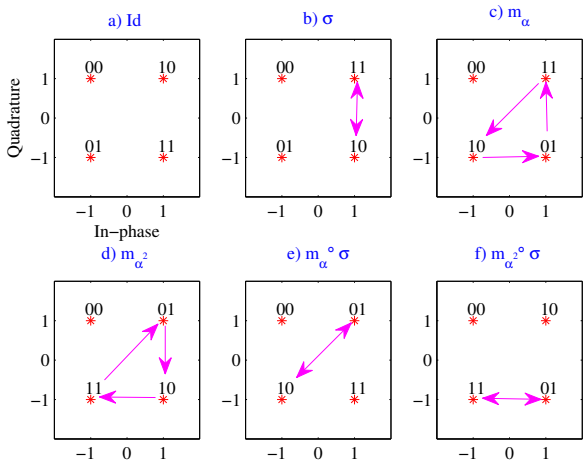


Figure 6.3 — Diagrammes de constellation de symboles de 4-QAM pour des fonctions du $\overline{\mathbb{MD}}$

6.3 Exemple de détection de défauts liés à une mauvaise interprétation des éléments de GF(2^m)

Un autre problème courant qui peut arriver est la mauvaise interprétation dans une machine de calcul qui traite des symboles d'un corps de Galois. Nous allons regarder, à titre d'exemple, l'impact de telles interprétations erronées sur la dimension de l'espace engendré par un code correcteur d'erreurs non-binaire travaillant dans ce même corps. Supposons que les données que nous devons manipulées ont été protégées par un code correcteur d'erreurs non-binaire et que le but est de mettre en oeuvre un décodeur pour ce code. Afin d'être en mesure de faire les calculs, il est nécessaire de connaître la représentation utilisée à la génération de ces données codées et également le polynôme primitif utilisé pour générer l'ensemble des éléments et pour effectuer les calculs dans le corps de Galois.

Un certain nombre de problème peuvent se produire au niveau du décodeur on ne connait pas la représentation et le polynôme primitif utilisés. Il s'agit ici d'un contexte non-coopératif qui est généralement connu en cryptanalyse ou lors de l'interception de communications. Ces problèmes peuvent également se poser si les concepteurs au niveau de l'émission ont utilisé une représentation non classique et que du coté décodeur une autre représentation est utilisée. Le rôle du récepteur est alors de détecter la représentation non classique utilisée à l'émission afin de s'adapter à cette représentation.

Dans un processus classique, les symboles d'un corps de Galois de caractéristique 2 (GF(2^m)) sont représentés par un regroupement de m bits. La représentation binaire classique des symboles correspond à avoir chaque bit pondéré par des puissances de 2 avec le bit de poids faible à droite et le bit de poids fort à gauche. Cependant, dans certaines implémentations, il est possible de rencontrer une autre représentation correspondant à mettre le bit de poids faible à gauche et le bit de poids fort à droite. Certains processeurs inversent totalement les bits où le zéro remplace le un et le un remplace le zéro ($0 \longleftrightarrow 1$). Il est également possible de prendre un polynôme primitif différent que celui utilisé classiquement. Cela engendre donc une incompatibilité au décodage. Nous allons donc illustrer les possibles problèmes de représentation sur ces cas :

1. Inversion des bits de poids faibles et des bits de poids forts,

2. Inversion totale de chaque bit (inversion de polarité) : $0 \longleftrightarrow 1$,
3. Changement du polynôme primitif.

Pour les cas 1 et 2, le tableau 6.2 nous permet de voir les changements de représentation engendrés par ces deux problèmes dans le cas de GF(2^2), ainsi que les fonctions de mapping-demapping associées à ces changements. Au regard de notre formalisme et des propriétés que nous en avons déduites, dans GF(2^2) le changement 1 correspondant à une inversion des poids forts et des poids faibles ne pourra pas être détecté, d'après le théorème 5.21, par le calcul de la dimension de l'espace engendré sans information à priori car la fonction de mapping-demapping correspondante appartient au sous-groupe universel $\overline{\mathbb{MD}}$ dans le cas de GF(2^2). Par contre, l'inversion de polarité ($0 \longleftrightarrow 1$) sera détectée si le mot $(\mathbf{1})_i$ composé que de 1 n'appartient pas aux mots de code. Le problème de changement du polynôme primitif ne peut jamais se produire pour la représentation dans GF(2^2) puisqu'il n'existe qu'un seul polynôme irréductible de degré 2 dans GF(2)[x].

Nous avons vu que certains défauts de la mauvaise représentation de symboles de GF(2^2) correspondant aux fonctions de mapping-demapping appartenant au groupe quasi-universel $\widetilde{\mathbb{MD}}$ peuvent être détectés. Cependant, nous ne pouvons pas simplement généraliser cette observation à des corps de Galois de dimension $2^m > 2^2$. Pour cela, nous allons voir si les défauts 1, 2 et 3 peuvent être détectés dans le cas du corps GF(2^3). Le changement de représentation engendré par les problèmes 1 et 2 ainsi que les fonctions de mapping-demapping correspondant à ce changement sont représentés sur le tableau 6.3. Le défaut 1 correspond à une fonction particulière d qui inverse les bits des poids forts et faibles pour les éléments de GF(2^3). D'après notre formalisme et plus précisément le théorème 5.21, cette fonction n'appartient pas au groupe quasi-universel si $m \neq 2$. Alors, il est possible de détecter le problème d'inversion de poids dans le cas de GF(2^3) pour certain espace du code lorsque la dimension du code ne sera pas conservée. Par contre, la fonction de mapping-demapping qui correspond au défaut 2 est une fonction de l'ensemble strictement quasi-universel. D'après la propriété présentée dans le théorème 5.19, si le mot $(\mathbf{1}_i)$ appartient à l'espace engendré par les mots de code, la dimension du code sera conservée, sinon cette dimension augmentera de un. Cette propriété peut être utilisée comme critère afin de détecter le problème 2 à travers l'observation du changement de la dimension de l'espace engendré par le code. Nous étudions maintenant l'impact du défaut 3 qui correspond au changement de polynôme primitif sur les représentations des élément de GF(2^3). Prenons l'exemple de polynôme primitif $P_2(X)$ considéré dans l'exemple de la section 5.5.1 afin de générer les symboles de GF(2^3) pour la représentation classique. Le tableau 5.4 nous permet de voir la modification de la représentation classique lors du changement de polynôme primitif $P_2(X)$ par $P_1(x)$. Au regard de notre formalisme et plus précisément le théorème 5.20, la fonction de mapping-demappping correspondant au changement de polynôme primitif n'appartient ni au sous-groupe universel $\overline{\mathbb{MD}}$ ni à l'ensemble strictement quasi-universel. Il est donc possible de détecter le défaut correspondant à cette fonction pour certain espace du code lorsque la dimension du code n'est pas conservée.

Représentations de la machine	Représentations classiques Bit de poids fort à gauche			Bit de poids fort à droite			Inversion de polarité $0 \longleftrightarrow 1$		
fonction de mapping-demapping	id			$m_\alpha \circ \sigma$			$id + \alpha^2$		
Format de la représentation	Puissance de α	Binaire	Entier	Puissance de α	Binaire	Entier	Puissance de α	Binaire	Entier
Valeur	0	(0 0)	0	0	(0 0)	0	α^2	(1 1)	3
	1	(0 1)	1	α	(1 0)	2	α	(1 0)	2
	α	(1 0)	2	1	(0 1)	1	1	(0 1)	1
	α^2	(1 1)	3	α^2	(1 1)	3	0	(0 0)	0

Tableau 6.2 — Exemple de représentations de symboles dans GF(2^2)

Représentation de la machine	Représentations classiques Bit de poids fort à gauche			Bit de poids fort à droite			Inversion de polarité $0 \longleftrightarrow 1$		
Fonction de mapping-demapping	id			Fonction non quasi-universelle			$id + \alpha^3$		
Format de la représentation	Puissance de α	Binaire	Entier	Puissance de α	Binaire	Entier	Puissance de α	Binaire	Entier
Valeur	0	(0 0 0)	0	0	(0 0 0)	0	α^5	(1 1 1)	7
	1	(0 0 1)	1	α^2	(1 0 0)	4	α^4	(1 1 0)	6
	α	(0 1 0)	2	α	(0 1 0)	2	α^6	(1 0 1)	5
	α^2	(1 0 0)	4	1	(0 0 1)	1	α^3	(0 1 1)	3
	$\alpha^3 = \alpha + 1$	(0 1 1)	3	α^4	(1 1 0)	6	α^2	(1 0 0)	4
	$\alpha^4 = \alpha^2 + \alpha$	(1 1 0)	6	α^3	(0 1 1)	3	1	(0 0 1)	1
	$\alpha^5 = \alpha^2 + \alpha + 1$	(1 1 1)	7	α^5	(1 1 1)	7	0	(0 0 0)	0
	$\alpha^6 = \alpha^2 + 1$	(1 0 1)	5	α^6	(1 0 1)	5	α	(0 1 0)	2

Tableau 6.3 — Exemple de représentations de symboles dans GF(2^3), pour un polynôme primitif $p(x) = x^3 + x + 1$

6.4 Impact des fonctions de mapping-demapping sur l'identification d'un code non-binaire

6.4.1 Impact des fonctions quasi-universelles

Dans cette section, nous étudions l'effet des fonctions de mapping-demapping appartenant au groupe quasi-universel sur l'identification aveugle des paramètres des codes non-binaires. Nous avons vu précédemment que ce groupe est subdivisé en deux sous-ensembles : le sous-ensemble universel et le sous-ensemble strictement quasi-universel. Nous avons étudié théoriquement les propriétés de chaque sous-ensemble. Nous avons montré que le premier sous-ensemble est un groupe caractérisé par la conservation de la dimension de l'espace engendré par tout code. D'après notre formalisme, le deuxième sous-ensemble ajoute 1 à la dimension du code si le mot composé uniquement de 1 n'est pas un mot du code. Par la suite, nous allons illustrer ces propriétés et voir leurs effets, en simulations, sur l'identification aveugle d'un code dans $\mathrm{GF}(2^m)$.

Nous rappelons que les fonctions quasi-universelles de mapping-demapping s'écrivent sous la forme $h = m_\lambda \circ \sigma^i + \mu$, avec $i \in [\![0, m]\!]$, $\lambda \in \mathrm{GF}(2^m)^*$ et $\mu \in \mathrm{GF}(2^m)$. Soit $\mathbf{r}$ une trame de mots de code non-binaire. Avant de transmettre les mots de code sur un canal non-bruité, une opération de mapping est appliquée afin d'affecter à chaque symbole envoyé un point ou un symbole de la constellation. Mais, en présence d'un demapping inconnu, l'opération inverse de mapping (demapping) ne permet pas d'obtenir les bons symboles codés. La fonction de mapping-demapping qui décrit ce défaut est la fonction h. Notre objectif est alors d'essayer d'identifier les paramètres du code utilisé à l'émission à partir des symboles $h(\mathbf{r})$ issus de l'opération de demapping. La figure 6.4 décrit le principe de transmission lorsque un code correcteur d'erreurs dans $\mathrm{GF}(2^m)$ et une fonction quasi-universelle de mapping-demapping sont utilisés.

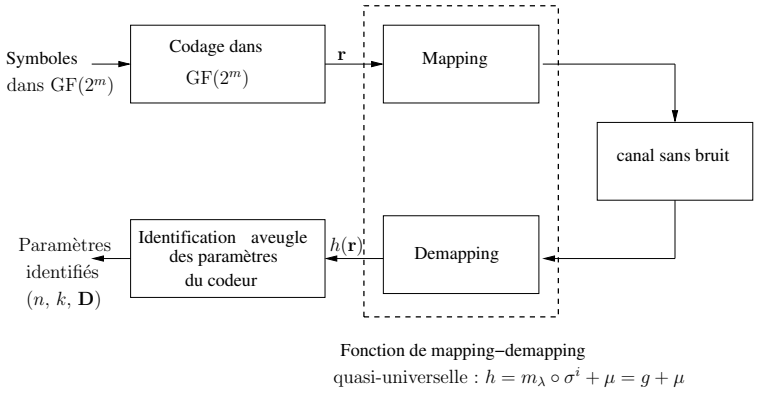


Figure 6.4 — Schéma de la chaîne de transmission pour des fonctions du groupe $\widetilde{\mathbb{MD}}$

Dans le cadre de cette étude, nous considérons un code LDPC(6,3) construit sur le corps de Galois $\mathrm{GF}(2^3)$ et défini par la matrice de contrôle de parité :

$$\mathbf{H} = \begin{pmatrix} 0 & 1 & 2 & 0 & 2 & 0 \\ 4 & 0 & 0 & 5 & 0 & 2 \\ 7 & 0 & 6 & 0 & 0 & 7 \end{pmatrix} \tag{6.1}$$

Pour identifier les paramètres de ce code dans le cas d'une transmission non-bruitée, nous appliquons la méthode du rang décrite dans le chapitre 3. Il est possible d'identifier une base du code dual à l'aide de l'algorithme 1. Les symboles codés sont convertis puis reconvertis en utilisant la fonction quasi-universelle équivalente h. Dans ce contexte, en utilisant l'équation 5.12, le nombre de fonctions quasi-universelles possibles est 168. Nous allons prendre quelques exemples de ces

fonctions pour voir leurs effets sur l'identification du code LDPC(6, 3).

6.4.1.1 Fonctions universelles

D'après le formalisme présenté dans le chapitre 5, une fonction universelle de mapping-demapping, notée g, s'écrit sous la forme : $g = m_\lambda \circ \sigma^i$, avec $i \in [\![0,3]\!]$ et $\lambda \in \mathrm{GF}(2^3)^*$. Le nombre de ces fonctions est 21. Prenons comme exemple les quatre fonctions : $g_1 = id$, $g_2 = \sigma$, $g_3 = m_\alpha$ et $g_4 = m_\alpha \circ \sigma$, où $\alpha \in \mathrm{GF}(2^3)$ est une racine du polynôme primitif $x^3 + x + 1$. Le tableau qui représente les résultats de transformation des éléments de $\mathrm{GF}(2^3)$ par ces fonctions est présenté dans l'annexe H. Nous allons identifier les paramètres du code LDPC(6,3) à partir des symboles $g(\mathbf{r})$.

Identification des paramètres n et k du code LDPC(6,3) dans GF(2^3)

Pour identifier les paramètres n et k, nous construisons des matrices $\mathbf{R}_l$, avec $l = 1, \cdots, 40$, à partir des symboles $g_i(\mathbf{r})$, $i = \{1, 2, 3, 4\}$, obtenus en sortie de demapping. Puis, nous calculons le rang de chaque $\mathbf{R}_l$ dans les quatre cas. Nous représentons sur la figure 6.5, les comportements du rang des matrices $\mathbf{R}_l$, pour les fonctions g_i.

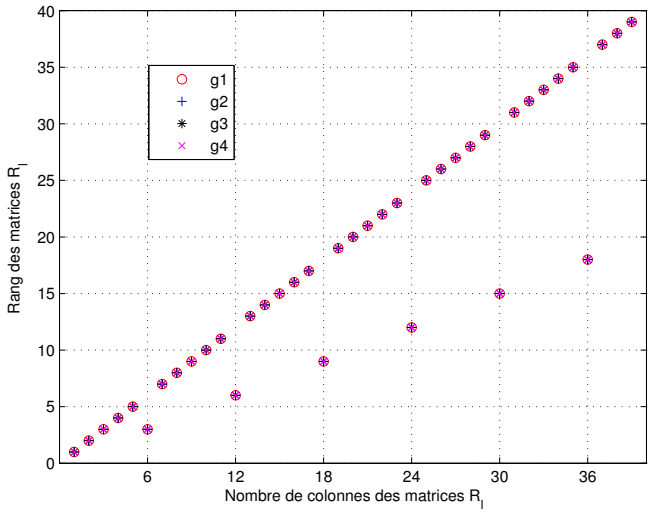


Figure 6.5 — Rang des matrices R_l pour les fonctions g_i et LDPC(6, 3)

D'après cette figure, il est possible de voir que pour toutes les fonctions g_i, le rang des matrices $\mathbf{R}_l$ possèdent bien les deux comportements habituels et les chutes de rang nous permettent d'estimer les bons paramètres du code. La propriété de conservation de la dimension du code pour des fonctions de mapping-demapping universelles est bien mise en évidence. Nous pouvons déduire que les problèmes de transmission engendrés par les fonctions universelles g_1, g_2, g_3 et g_4 n'ont aucune influence sur l'identification des paramètres n et k du code. La question qui se pose alors est de savoir s'il est possible d'identifier une base du code dual ?

Identification d'une base du code dual du code LDPC(6,3) dans GF(2^3)

Nous allons présenter les résultats de l'identification aveugle d'une base du code dual du code LDPC$(6, 3)$ lorsque des fonctions universelles de mapping-demapping g_i sont appliquées.

- Pour $g_1 = id$: c'est le cas parfait où les opérations de mapping-demapping sont complémentaires. Dans ce cas, la base duale identifiée sous forme matricielle, notée $\mathbf{D}$, est une matrice systématique donnée par :

$$\mathbf{D} = \begin{pmatrix} 7 & 0 & 2 & 1 & 0 & 0 \\ 0 & 5 & 1 & 0 & 1 & 0 \\ 1 & 0 & 5 & 0 & 0 & 1 \end{pmatrix}$$

  Cette matrice correspond à une matrice équivalente de la matrice $\mathbf{H}$ (donnée par l'équation 6.1).

- Pour $g_2 = \sigma$: l'effet de la fonction du Frobenius σ apparait clairement dans l'identification d'une base dual du code. La base identifiée sous forme matricielle, notée $\mathbf{D}_1$ est donnée par :

$$\mathbf{D}_1 = \begin{pmatrix} 3 & 0 & 4 & 1 & 0 & 0 \\ 0 & 7 & 1 & 0 & 1 & 0 \\ 1 & 0 & 7 & 0 & 0 & 1 \end{pmatrix}$$

  On peut constater que cette matrice est une transformation de la matrice $\mathbf{D}$ par le Frobenius, c'est-à-dire $\mathbf{D}_1 = \sigma(\mathbf{D})$. On peut en déduire que pour la fonction g_2, il est possible d'identifier les bons paramètres n et k du code, mais il est impossible d'identifier une base correcte du code dual. Pour cela, il faut identifier la bonne fonction de demapping afin de reconnaitre le code correcteur d'erreurs non-binaire.

- Pour $g_3 = m_\alpha$: la base du code identifiée dans ce cas, notée $\mathbf{D}_2$, correspond à la base $\mathbf{D}$ identifiée dans le cas $g_1 = id$, c'est-à-dire $\mathbf{D}_2 = \mathbf{D}$. La multiplication par un élément non-nul de $\mathrm{GF}(2^3)$ ne semble avoir aucune influence sur l'identification des paramètres du code, ainsi que sur la base du code dual. Dans ce cas, on a vérifié la propriété démontrée théoriquement dans notre formalisme qui énonce que la multiplication par un scalaire non-nul conserve la dimension du code et elle conserve également toutes les propriétés du code (voir le théorème 5.12).

- Pour $g_4 = m_\alpha \circ \sigma$: en utilisant l'algorithme d'identification d'une base du code dual, la base identifiée, notée $\mathbf{D}_3$, s'écrit en fonction de g_2 : $\mathbf{D}_3 = g_2(\mathbf{D}) = \sigma(\mathbf{D}) = \mathbf{D}_1$.

A partir des exemples présentées pour les fonctions universelles, on peut conclure qu'il est possible d'identifier les bons paramètres n et k du code non-binaire dans le cas de l'existence des défauts de transmission correspondant aux fonctions universelles, mais il existe certaines fonctions qui sont sous la forme de $m_\lambda \circ \sigma^i$, avec $i \in \{1, \cdots, m-1\}$ qui ne permettent pas d'identifier une base correcte du code dual.

6.4.1.2 Fonctions strictement quasi-universelles

Une fonction strictement quasi-universelle, notée h, est la composition d'une fonction universelle g et d'une translation par μ sur $\mathrm{GF}(2^m)$. Dans le cas du corps $\mathrm{GF}(2^3)$, le nombre de ces fonctions est $168 - 21 = 147$. Prenons un exemple de fonction $h = g_4 + \alpha = m_\alpha \circ \sigma + \alpha$, où α est une racine du polynôme primitif $x^3 + x + 1$. Nous essayons maintenant d'identifier les paramètres et une base du code dual du code LDPC$(6, 3)$ lorsque la fonction h est appliquée à la trame de mots de code $\mathbf{r}$.

Les symboles $h(\mathbf{r})$ obtenus en sortie du demapping sont réorganisés pour former des matrices $\mathbf{R}_l$ dont le nombre de colonne l varie entre 1 et 40. Le rang calculé pour toutes les matrices $\mathbf{R}_l$ est représenté en fonction de l sur la figure 6.6.

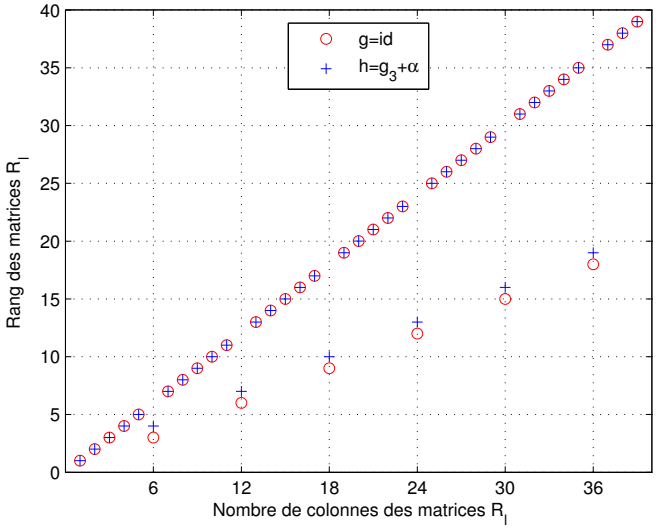


Figure 6.6 — Rang des matrices R_l pour $h = g_3 + \alpha$ et LDPC(6, 3)

On peut observer deux comportements possibles du rang lorsque la fonction h est utilisée. L'équation de la droite qui décrit le comportement des chutes de rang est donnée par :

$$rang(\mathbf{R}_l) = \frac{3}{6} \cdot l + 1$$

D'après la forme de cette équation, le code identifié n'est pas un code en bloc. Elle semble correspondante à celle d'un code convolutif de paramètres : $\tilde{n} = 6$, $\tilde{k} = 3$ et $\mu^{\perp} = 1$. Cependant, les paramètres identifiés $\tilde{n}$ et $\tilde{k}$ correspondent bien aux paramètres du code LDPC utilisé à l'émission. On peut remarquer que la fonction strictement quasi-universelle h augmente de 1 le rang des matrices $\mathbf{R}_l$. Ce résultat confirme la propriété de ces fonctions énoncée dans le théorème 5.19. D'après ce théorème, l'ajout de 1 à la dimension du code est justifié par la non appartenance du mot composé uniquement de 1 à l'espace vectoriel engendré par le code.

Puisque les paramètres identifiés ne correspondent pas aux paramètres d'un code en bloc, il est impossible d'identifier la bonne base du code dual.

6.4.2 Impact des fonctions non quasi-universelles

D'après notre formalisme, il existe 40152 fonctions non quasi-universelles dans le cas du corps GF(2^3). Dans le cadre de notre étude, nous nous sommes intéressés aux deux fonctions particulières qui sont la fonction correspondant au changement du polynôme primitif et la fonction correspondant à l'inversion des poids forts et faibles. Nous allons analyser l'influence de ces deux fonctions sur l'identification aveugle des codes correcteur d'erreurs non-binaires.

6.4.2.1 Changement du polynôme primitif

Nous avons vu dans le chapitre 3 l'impact du changement du polynôme primitif sur l'identification des paramètres des codes non-binaires. Nous avons montré deux comportements possibles de rang des matrices $\mathbf{R}_l$ construites à partir des données reçues. Pour certains codes, le changement de polynôme primitif engendre un rang plein pour toutes les matrices $\mathbf{R}_l$, ce qui ne permet pas de détecter l'existence d'un code. Par contre, il existe des codes où le changement de polynôme primitif peut engendrer des chutes de 1 ce qui permet de détecter l'existence d'un code. Dans le chapitre 3, nous n'avons pas justifié l'existence de ce comportement dans le cas d'un mauvais poly-

nôme primitif. Mais, avec le développement de notre formalisme, il est possible de répondre à cette question. Dans ce contexte, nous reprenons l'exemple, présenté dans la sous section 3.4.2.2, du code LDPC$(12, 6)$ dans GF(2^3) défini par la matrice de contrôle de parité $\mathbf{H}_2$:

$$\mathbf{H}_2 = \begin{pmatrix} 5 & 0 & 2 & 7 & 2 & 7 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 1 \\ 0 & 0 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 7 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6 & 7 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 5 & 5 & 0 & 0 & 0 & 0 \\ 7 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 7 & 5 \end{pmatrix} \tag{6.2}$$

Le polynôme primitif utilisé lors de l'émission est $x^3 + x + 1$. A la réception, un autre polynôme primitif donné par $x^3 + x^2 + 1$ a été utilisé.

D'après la figure 6.7 qui représente le rang des matrices $\mathbf{R}_l$ pour le code LDPC$(12, 6)$ dans GF(2^3), les paramètres du code identifiés dans le cas du changement de polynôme primitif est : $\tilde{n} = n = 12$ et $\tilde{k} = 11$. Nous remarquons que la bonne taille du code est identifiée, mais pas la taille des mots d'information k.

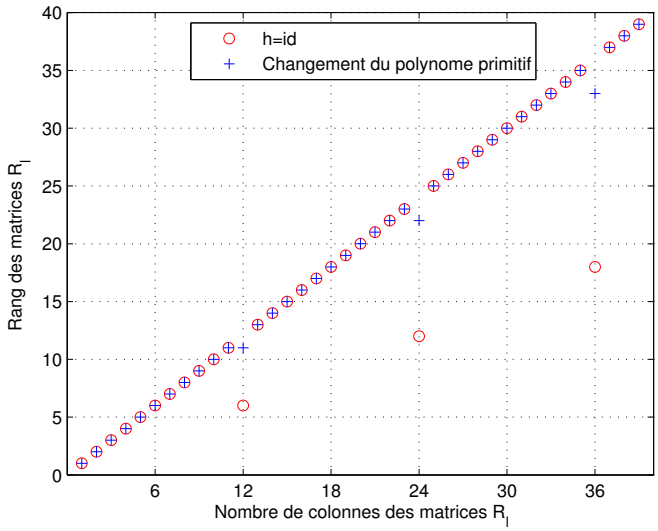


Figure 6.7 — Rang des matrices R_l pour LDPC(12,6) dans le cas d'un mauvais polynôme primitif

Nous présentons maintenant le résultat de l'identification d'une base du code dual dans le cas de l'utilisation d'un mauvais polynôme primitif. La base identifiée, notée $\mathbf{D}'$, qui contient une seule relation de parité est donnée par :

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

On peut vérifier que la relation de parité identifiée appartient à une base du code dual du code LDPC$(12, 6)$. A partir de ce résultat, nous déduisons que la fonction correspondant au changement de polynôme primitif a conservé une relation de parité du poids 2. C'est pour cette raison, nous voyons des chutes de rang de 1 dans le cas d'un mauvais polynôme primitif.

6.4.2.2 Inversion des poids forts et faibles

Pour illustrer l'impact de la fonction de mapping-demapping correspondant à l'inversion des poids, nous prenons trois codes :

- LDPC$(6,3)$ dans GF(2^3) défini par la matrice de contrôle de parité $\mathbf{H}$ donnée par l'équation (6.1),
- LDPC$(12,6)$ dans GF(2^3) défini par la matrice de contrôle de parité $\mathbf{H}_2$ donnée par l'équation (6.2),
- LDPC$(6,3)$ dans GF(2^2) défini par la matrice de contrôle de parité $\mathbf{H}'$:

$$\mathbf{H}' = \begin{pmatrix} 0 & 1 & 1 & 0 & 2 & 0 \\ 3 & 0 & 0 & 3 & 0 & 1 \\ 2 & 0 & 2 & 0 & 0 & 1 \end{pmatrix}$$

Pour ces trois codes, nous construisons des matrices $\mathbf{R}_l$ à partir des symboles reçus convertis et reconvertis par la fonction correspondant à l'inversion des poids forts et de poids faibles.

Code LDPC$(6,3)$ dans GF(2^3)

Nous représentons sur la figure 6.8 le rang des matrices $\mathbf{R}_l$ pour le code LDPC$(6,3)$ dans GF(2^3). D'après cette figure, nous constatons que le rang de toutes les matrices $\mathbf{R}_l$ est plein lorsque la fonction de mapping demapping correspondant à l'inversion des poids est appliquée aux symboles reçus. Dans ce cas, il est impossible de détecter le code.

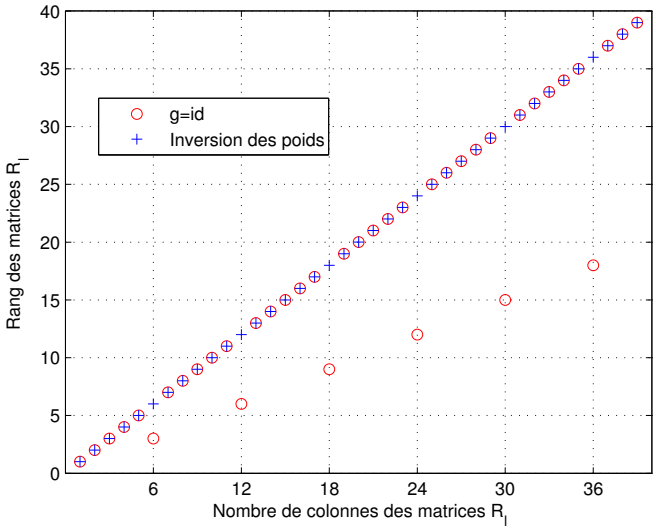


Figure 6.8 — Rang des matrices R_l pour le code LDPC(6,3) dans le cas de l'inversion des poids dans GF(2^3)

Code LDPC$(12,6)$ dans GF(2^3)

Sur la figure 6.9, nous représentons le rang des matrices $\mathbf{R}_l$ pour le code LDPC$(12,6)$ dans GF(2^3) dans le cas de l'utilisation de la fonction correspondant à l'inversion des poids. Nous pouvons remarquer qu'il existe des chutes de rang de 1 dans quelques matrices $\mathbf{R}_l$. En utilisant ces chutes, on peut identifier un code de paramètres : $\tilde{n} = n = 12$ et $\tilde{k} = 11$. On voit que ce code ne correspond pas au code utilisé à l'émission. Dans ce contexte, on va vérifier s'il existe une relation de parité qui est conservée par la fonction de mapping-demapping d'inversion des poids forts et faibles. En appliquant notre algorithme d'identification d'une base du code dual, la base identifiée composée d'une seule relation de parité est la base $\mathbf{D}'$ qui a été identifiée dans le cas du changement de

polynôme primitif. Donc, on peut justifier l'existence des chutes de rang de 1 par la conservation d'une relation de parité de poids 2.

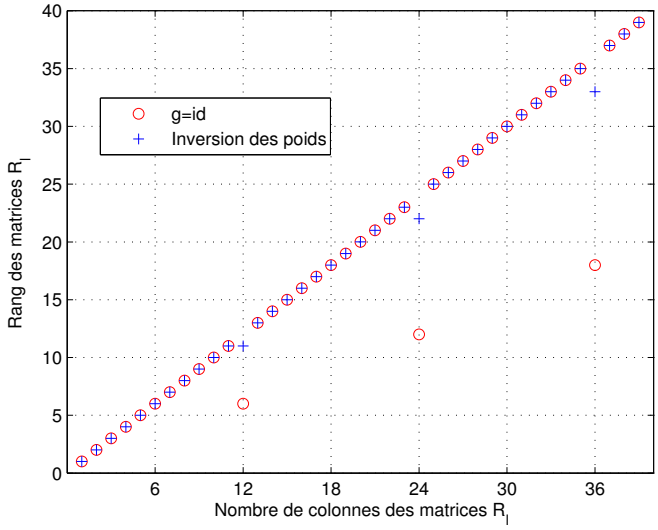


Figure 6.9 — Rang des matrices R_l pour le code LDPC(12,6) dans le cas d'une inversion de poids dans GF(2^3)

Code LDPC$(6,3)$ dans GF(2^2)

Nous représentons sur la figure 6.10 le rang des matrices $\mathbf{R}_l$ pour le code LDPC$(6,3)$ dans GF(2^2) dans le cas de l'utilisation de la fonction d'inversion des poids forts et des poids faibles. On peut remarquer l'existence des chutes de rang dans quelques matrices $\mathbf{R}_l$ qui permettent d'identifier les bons paramètres du code.

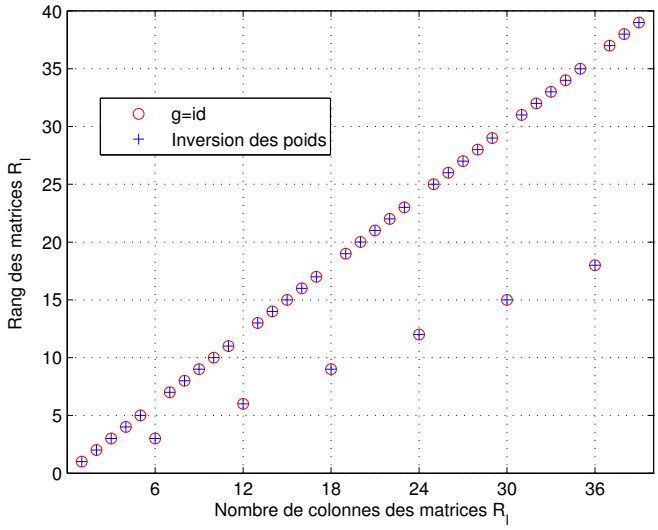


Figure 6.10 — Rang des matrices R_l pour le code LDPC(6,3) dans le cas d'une inversion de poids dans GF(2^2)

L'identification d'une base du code dual nous donne une base, notée $\mathbf{D}_i$, qui s'écrit sous la forme

matricielle :

$$\mathbf{D}_i = \begin{pmatrix} 3 & 0 & 2 & 1 & 0 & 0 \\ 0 & 2 & 2 & 0 & 1 & 0 \\ 3 & 0 & 3 & 0 & 0 & 1 \end{pmatrix}$$

Cette matrice correspond au Frobenius de la base $\mathbf{D}$ identifiée dans le cas de $g_1 = id$ qui est donnée par :

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

Donc, nous pouvons démontrer ici le théorème 5.21 qui énonce que la fonction correspondant à l'inversion des poids est quasi-universelle, plus précisément universelle dans le cas du corps $\mathrm{GF}(2^2)$. On peut déduire également que cette fonction peut s'écrire en fonction du Frobenius puisque la base identifiée dans cet exemple est le Frobenius de la bonne base du code dual. D'après le tableau 5.1, la fonction correspondante à une inversion des poids dans cet exemple est $m_\alpha \circ \sigma$.

6.5 Conclusion

Dans la première partie de ce chapitre, nous avons montré que certains problèmes courant de transmission lors de l'utilisation d'une modulation de type QAM pour des données protégées par un code correcteur d'erreurs non-binaire où les symboles sont adaptés à la taille de la modulation, peuvent être détectés et corrigés sous certaines conditions. Au regard des propriétés des fonctions de mapping-demapping correspondant à ces problèmes, la détection de ces problèmes peut être réalisée seulement en évaluant la dimension du code au niveau du récepteur. La détection de problèmes de représentation des symboles d'un corps de Galois dans un processus de calculs a également été abordée et nous avons montré que certains problèmes peuvent être détectés comme l'utilisation d'un mauvais polynôme primitif ou de l'inverse de poids sur les bits.

Dans la deuxième partie de ce chapitre, nous avons étudié l'impact de l'utilisation des fonctions quasi-universelles et non quasi-universelles sur l'identification aveugle des codes correcteur d'erreurs non-binaires. Lors de l'utilisation de fonctions universelles, nous avons montré qu'il est possible d'identifier les paramètres du code lorsque ces fonctions appartiennent au groupe universel. Nous avons montré également qu'avec ces fonctions universelles qu'il est possible d'identifier une base du code dual lorsque ces fonctions correspondent à une multiplication par un élément non-nul du corps de Galois, sinon on identifie une base qui correspond à une puissance du Frobenius de la bonne base du code dual. Par contre, si les fonctions de mapping-demapping appartiennent à l'ensemble strictement quasi-universel, il est possible de détecter l'existence d'un code mais pas d'identifier les paramètres du code utilisé à l'émission. Pour les fonctions non quasi-universelles, nous nous sommes intéressés à deux fonctions particulières qui correspondent au changement de polynôme primitif et à une inversion des poids forts et faibles. Lors de l'utilisation d'un mauvais polynôme primitif, nous avons détecté l'existence du code et nous avons vu des chutes de rang de 1 lorsque une relation de parité de poids 2 appartenant à la bonne base du code dual est identifiée. Dans le cas de la fonction d'inversion des poids, nous avons obtenu les mêmes résultat dans le cas du corps $\mathrm{GF}(2^3)$. Dans le cas du corps $\mathrm{GF}(2^2)$, nous avons montré par un exemple que la fonction correspondant à une inversion des poids conserve la dimension du code et elle fait partie du groupe universel.

Conclusion

Afin d'améliorer la qualité des systèmes de transmission numériques, les codes correcteurs d'erreurs sont fréquemment utilisés. Ces codes sont conçus pour obtenir une bonne immunité contre les erreurs engendrées par les canaux de transmission en introduisant de la redondance dans les données d'information. Dans ce mémoire, nous nous sommes concentrés sur des codes correcteurs d'erreurs binaires et non-binaires. Les codes correcteurs d'erreurs non-binaires abordés dans cette thèse traitent des données appartenant aux corps de Galois GF(2^m), où $m > 1$. La structure du corps de Galois a fait l'objet d'une étude dans le chapitre 1. Dans ce chapitre, nous avons rappelé quelques notions sur les différentes structures algébriques, en particulier la structure du corps de Galois. Ces notions nous ont permis d'obtenir les paramètres indispensables pour générer les éléments de ce corps. Nous avons montré que la construction usuelle d'un corps de Galois est basée sur la connaissance de son cardinal 2^m et sur un polynôme primitif de degré m.

Afin d'être en mesure d'effectuer l'opération de décodage, le récepteur a besoin de connaître les paramètres de codage utilisés à l'émission. Les technologies utilisées actuellement sont basées sur l'entente préalable entre l'émetteur et le récepteur sur le schéma de codage et ses paramètres. Ces technologies sont fonctionnelles, mais elles ne sont plus adaptées au développement des nouveaux schémas de codage plus performants et à la prolifération des nouvelles normes et standards de communication. Les systèmes radio cognitifs fournissent une solution pertinente à ce problème à travers la conception de récepteurs intelligents qui seront capables d'identifier en aveugle les paramètres du système de codage avec la seule connaissance des données reçues. Jusqu'à présent, les travaux sur la thématique de l'identification aveugle des paramètres du bloc de codage canal ont été, pour la plupart, limités à des codes binaires. Dans le cadre de nos travaux, nous nous sommes donc intéressés à l'identification aveugle des codes correcteurs d'erreurs non-binaires. Pour traiter cette problématique, nous avons réalisé, dans le chapitre 2, une étude théorique sur les propriétés des codes abordés dans nos travaux afin d'obtenir les paramètres indispensables permettant de les identifier au niveau du récepteur. Parmi les codes dans GF(2^m), les codes LDPC non-binaires ont montré récemment de meilleurs performances par rapport aux codes LDPC binaires. Dans ce mémoire, nous nous sommes intéressés à ces codes, mais également aux codes Reed-Solomon et aux codes convolutifs non-binaires.

Dans le chapitre 3, nous avons fait l'hypothèse que les données reçues n'étaient pas entachées d'erreurs afin d'évaluer la faisabilité d'une telle identification avant d'envisager le cas où il y a des erreurs. Notre idée a été d'adapter la méthode existante d'identification des codes binaires basée sur le critère du rang au cas des codes non-binaires. Le principe de cette méthode consiste à utiliser les chutes de rang des matrices construites à partir des données reçues pour identifier les paramètres du code. Cependant, nous avons constaté que ces chutes de rang ont été exploitées sans justification théorique par la majorité des méthodes proposées pour identifier en aveugle les paramètres de codes binaires. Nous avons donc effectué une étude théorique approfondie afin de justifier ce comportement. Ensuite, nous avons mis en évidence l'impact de la matrice génératrice

sur certaines chutes de rang. Notre étude a aussi donné une explication à l'existence des chutes de rang multiples qui dépendent des codes ayant des paramètres spécifiques. Dans ce contexte, nous avons proposé une expression générale pour le calcul du rang afin de traiter les comportements possibles du critère du rang, c'est-à-dire les chutes multiples et les chutes de rang permettant l'identification des paramètres du code.

Nous avons ensuite généralisé l'algorithme d'identification des paramètres des codes binaires basé sur le critère du rang au cas des codes correcteurs d'erreurs dans GF(2^m). Nous avons étudié l'influence d'une mauvaise estimation des paramètres du corps de Galois sur l'identification des paramètres des codes convolutifs, des codes LDPC et des codes de Reed-Solomon. Cette étude a démontré la pertinence de l'algorithme d'identification des codes dans GF(2^m) sous l'hypothèse que les paramètres du corps de Galois (m et le polynôme primitif,...) utilisés à l'émission soient connus ou correctement identifiés par le récepteur. En supposant que les paramètres du code non-binaire ont été identifiés, nous avons aussi présenté un algorithme pour identifier en aveugle une base du code dual dans des conditions de transmission parfaite.

Dans le chapitre 4, nous avons traité le problème plus ardu de l'identification des paramètres des codes non-binaires lorsque les données codées sont entachées d'erreurs. Dans nos études théoriques, nous avons considéré un canal symétrique non-binaire. Nous nous sommes intéressés à l'identification de la taille des mots de code n et d'une base du code dual. Pour identifier la taille n, nous avons tout d'abord généralisé une technique existante appliquée aux codes binaires pour le cas des codes non-binaires. Cette technique est basée sur la recherche des colonnes presque dépendantes dans les matrices construites à partir des données reçues. Nous avons montré que cette technique ne peut fonctionner qu'en connaissant la probabilité d'erreur du canal. De ce fait, nous avons proposé deux nouvelles techniques d'identification de la taille n qui s'avèrent plus robustes et ne nécessitent pas en entrée la connaissance de la probabilité d'erreur du canal. Nous avons comparé les performances de détection des trois méthodes proposées et montré que les deux dernières sont meilleures en termes de taux de détection. Dans l'étude des performances, nous avons considéré différents types de canaux : un canal non-binaire symétrique, un canal gaussien et un canal de Rayleigh à trajets multiples.

Pour identifier une base du code dual, nous avons supposé que la taille du code n et la taille des mots d'information k sont connues. Nous avons généralisé une technique existante pour les codes binaires basée sur l'utilisation d'un démodulateur à décision ferme au contexte des codes non-binaires. Ensuite, nous avons amélioré cette technique en considérant une démodulation à décision souple afin d'exploiter l'information souple à travers le logarithme du rapport de vraisemblance (LRV). Cette information nous a permis de détecter les mots de code les moins entachés d'erreurs afin de les réorganiser sous la forme d'une matrice pouvant être utilisée par la suite dans l'algorithme d'identification. Une deuxième idée d'amélioration a été d'introduire un processus itératif dans l'algorithme classique. Dans notre processus, nous avons utilisé un algorithme de décodage itératif qui échange des informations à chaque itération avec l'algorithme d'identification des équations de parité. En effet, l'algorithme d'identification utilise les informations sur les fiabilités des mots de code corrigés, fournies par le décodeur, afin d'identifier les équations de parité qui vont être à leur tour utilisées par le décodeur pour corriger les mots de code reçus. Cependant, l'algorithme d'identification peut trouver un excès de relations de parité, lorsque le nombre de relations est supérieur à $n-k$. Dans ce cas, le décodeur peut introduire plus d'erreurs dans les mots de code reçus. Pour résoudre ce problème, nous avons modifié l'algorithme classique de décodage afin qu'il prenne en compte les probabilités de fiabilité des équations de parité estimées. Le calcul des probabilités de fiabilité des équations de parité nous a permis d'éliminer les équations de parité les moins fiables et de ne conserver que $n-k$ équations. L'étude des performances de détection des algorithmes proposés pour identifier en aveugle les $n-k$ équations de parité a montré une amélioration significative des

probabilités de détection obtenues grâce à la mise en oeuvre de l'algorithme itératif utilisant un algorithme de décodage à décision souple.

En parallèle des études détaillées précédemment, nous avons également proposé, dans le chapitre 5, un formalisme mathématique afin d'étudier l'impact des fonctions de mapping-demapping sur les calculs d'algèbre linéaire dans un corps de Galois effectués dans les blocs d'une chaîne de transmission pour les communications numériques. En effet, l'opération de mapping consiste à associer à un élément du corps de Galois une représentation spécifique. Pour récupérer cet élément à partir de sa représentation, il est nécessaire d'appliquer l'opération complémentaire du mapping qui est le demapping. D'un point de vue mathématique, la composition de ces deux opérations doit conduire à l'opérateur identité.

Dans certains cas, la fonction de mapping-demapping peut ne pas être égale à à l'opérateur identité. cela nous a amenés à nous poser la question de l'impact des différentes fonctions de mapping-demapping sur les calculs d'algèbre linéaire et sur leurs propriétés dans le corps de Galois utilisé. Dans un système de communications numériques, les problèmes de synchronisation peuvent produire involontairement ce phénomène, comme des rotations de constellations des symboles reçus. C'est également le cas lors d'une inversion de polarité sur la voie en phase et/ou en quadrature. Ils peuvent aussi être liés à des domaines d'applications spécifiques tels que la surveillance du spectre ou l'interception des communications. Dans de tels contextes, le récepteur ne connaît pas le mapping utilisé à l'émission.

Dans le cadre de cette thèse, nous nous sommes intéressés à la détection et l'identification de codes correcteurs d'erreurs. Ils introduisent de la redondance dans le train de données, ce qui conduit à des relations linéaires liant les éléments de chaque mot de code. Les codes, des sous-espaces vectoriels, ont des dimensions spécifiques qui sont égales à la taille de la partie informative des mots de code. Dans cette situation, la connaissance de l'influence de la fonction de mapping-demapping sur les propriétés de l'espace vectoriel engendré par le code peut éventuellement fournir des informations utiles et conduire à un critère permettant de détecter les défauts ou les modifications durant le processus de transmission. Avec l'aide de ce formalisme, nous avons présenté et étudié les caractéristiques de deux ensembles de fonctions de mapping-demapping. Le premier ensemble est un groupe que nous avons choisi d'appeler groupe universel. Ce groupe est caractérisé par la conservation de la dimension de tout espace vectoriel. Le deuxième ensemble appelé ensemble strictement quasi-universel qui dépend du code et de l'application linéaire utilisés conserve la dimension de certains espaces vectoriels et augmente de 1 les dimensions des autres codes linéaires. Les deux ensembles forment un groupe, le groupe quasi-universel.

Nous avons aussi étudié des cas particuliers de fonctions de mapping-demapping correspondant au changement de polynôme primitif et à l'inversion des bits de poids fort et de poids faible. Pour la fonction correspondant au changement de polynôme primitif, nous avons montré que cette fonction ne conserve généralement pas la dimension de tout code. De même pour la fonction d'inversion des poids, nous avons montré qu'elle n'est pas quasi-universelle sauf pour le corps $\mathrm{GF}(2^2)$. Nous avons aussi évalué théoriquement le nombre de fonctions de mapping-demapping qui sont capables de conserver toute relation de parité de poids 2. Cette étude permet de donner une idée des capacités de détection d'un code quelconque lorsque la fonction de mapping-demapping est inconnue et n'appartient pas au groupe quasi-universel. Dans le cas des fonctions du groupe quasi-universel, la détection de codes est toujours possible, mais l'identification des paramètres peut être liée au code utilisé.

Dans le chapitre 6, nous avons à titre d'exemple exploité les propriétés fournies par notre formalisme afin de détecter et de corriger certains défauts de transmission, telle que l'inversion de polarité

sur les voies en phase et en quadrature. Puis, nous avons étudié l'impact de certaines fonctions de mapping-demapping sur l'identification aveugle des paramètres des codes correcteurs d'erreurs non-binaires. Nous avons montré théoriquement et par simulation qu'il est possible d'identifier les paramètres du code lorsqu'une fonction de mapping-demapping appartenant au sous-groupe universel est utilisée.

En conclusion, les travaux de recherche liés à cette thèse ont permis d'apporter plusieurs contributions originales, parmi lesquelles les plus importantes sont listées ci-dessous :

- Mise en oeuvre d'une méthode d'identification aveugle des paramètres de codes correcteurs d'erreurs dans $\mathrm{GF}(2^m)$ dans le cas d'une transmission non-bruitée, et mise en évidence de la pertinence de cette méthode lorsque les paramètres du corps de Galois utilisés à l'émission sont connus à la réception.
- Justification de l'utilisation des chutes de rang maximales pour identifier les paramètres d'un code au travers d'une étude théorique approfondie et démonstration de l'existence de codes à paramètres spécifiques qui peuvent générer des chutes multiples, ainsi qu'une proposition de formule générale du calcul du rang qui prend en considération les différentes chutes de rang étudiées.
- Trois nouveaux algorithmes d'identification de la taille des mots de code non-binaires lorsque les données reçues sont entachées d'erreurs.
- Implémentation d'un algorithme itératif à décision souple d'identification aveugle d'une base du code dual pour les codes binaires et non-binaires.
- Un nouveau formalisme mathématique pour étudier l'impact des fonctions de mapping-demapping sur la manipulation des données d'un corps de Galois dans le cas des codes correcteurs d'erreurs non-binaires et exploitation de ce formalisme pour détecter et corriger quelques défauts de transmission comme la mauvaise synchronisation.

Dans nos futurs travaux, nous envisageons de justifier le problème d'apparition des chutes de rang lors de l'utilisation d'un sous-corps supposé identifié par le récepteur au lieu du bon corps. Dans le cas de l'identification des paramètres des codes non-binaires, nous souhaiterions développer une méthode d'identification de la taille des mots d'information k et de la matrice génératrice et généraliser la méthode d'identification à décision souple d'une base du code dual. Il serait aussi intéressant comme perspective d'exploiter notre formalisme sur les fonctions de mapping-demapping afin d'identifier en aveugle les paramètres des codes correcteurs d'erreurs non-binaires pour des transmissions utilisant des modulations à grand nombre d'états, avec les fonctions de mapping inconnues à la réception.

ANNEXE

A Différents types de polynômes dans GF(2^m)

Le but de cette annexe est de présenter différents types de polynômes dans les corps de Galois GF(2^2), GF(2^3), GF(2^4) et GF(2^5).

GF(2^m)	**Polynômes minimaux de GF(2)[x]**	**Polynômes irréductibles de GF(2)[x] de degré m**	**Polynômes primitifs de GF(2)[x] de degré m**
GF(2^2)	x $x+1$ x^2+x+1	x^2+x+1	x^2+x+1
GF(2^3)	x $x+1$ x^3+x+1 x^3+x^2+1	x^3+x+1 x^3+x^2+1	x^3+x+1 x^3+x^2+1
GF(2^4)	x $x+1$ x^2+x+1 x^4+x+1 x^4+x^3+1 $x^4+x^3+x^2+x+1$	x^4+x+1 x^4+x^3+1 $x^4+x^3+x^2+x+1$	x^4+x+1 x^4+x^3+1
GF(2^5)	x $x+1$ x^5+x^2+1 x^5+x^3+1 $x^5+x^3+x^2+x+1$ $x^5+x^4+x^2+x+1$ $x^5+x^4+x^3+x+1$ $x^5+x^4+x^3+x^2+1$	x^5+x^2+1 x^5+x^3+1 $x^5+x^3+x^2+x+1$ $x^5+x^4+x^2+x+1$ $x^5+x^4+x^3+x+1$ $x^5+x^4+x^3+x^2+1$	x^5+x^2+1 x^5+x^3+1 $x^5+x^3+x^2+x+1$ $x^5+x^4+x^2+x+1$ $x^5+x^4+x^3+x+1$ $x^5+x^4+x^3+x^2+1$

ANNEXE

B Tableaux d'addition et de multiplication dans GF(2^2) et dans GF(2^3)

Cette annexe présente les tableaux d'addition et de multiplication dans GF(2^2) et dans GF(2^3).

Dans le tableau ci-dessous, nous présentons les deux formats de représentation des éléments du corps GF(2^3) en utilisant le polynôme primitif $x^3 + x + 1$ qui a une racine α.

Puissance de α	Entier
0	0
1	1
α	2
α^2	4
α^3	3
α^4	6
α^5	7
α^6	5

- Tableau d'addition pour le corps GF(2^2) généré par l'élément primitif α, une racine du polynôme primitif $x^2 + x + 1$.

+		0 0	1 1	α 2	α^2 3
0	0	0	1	2	3
1	1	1	0	3	2
α	2	2	3	0	1
α^2	3	3	2	1	0

- Tableau de multiplication pour le corps GF(2^2) généré par l'élément primitif α, une racine du polynôme primitif $x^2 + x + 1$.

*		0 0	1 1	α 2	α^2 3
0	0	0	0	0	0
1	1	0	1	2	3
α	2	0	2	3	1
α^2	3	0	3	1	2

- Tableau d'addition pour le corps GF(2^3) généré par l'élément primitif α, une racine du polynôme primitif $x^3 + x + 1$.

$+$		0	1	α	α^3	α^2	α^6	α^4	α^5
		0	1	2	3	4	5	6	7
0	0	0	1	2	3	4	5	6	7
1	1	1	0	3	2	5	4	7	6
α	2	2	3	0	1	6	7	4	5
α^3	3	3	2	1	0	7	6	5	4
α^2	4	4	5	6	7	0	1	2	3
α^6	5	5	4	7	6	1	0	3	2
α^4	6	6	7	4	5	2	3	0	1
α^5	7	7	6	5	4	3	2	1	0

- Tableau de multiplication pour le corps GF(2^3) généré par l'élément primitif α, une racine du polynôme primitif $x^3 + x + 1$.

$*$		0	1	α	α^3	α^2	α^6	α^4	α^5
		0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0	0
1	1	0	1	2	3	4	5	6	7
α	2	0	2	4	6	3	1	7	5
α^3	3	0	3	6	5	7	4	1	2
α^2	4	0	4	3	7	6	2	5	1
α^6	5	0	5	1	4	2	7	3	6
α^4	6	0	6	7	1	5	3	2	4
α^5	7	0	7	5	2	1	6	4	3

ANNEXE C

Démonstrations mathématiques des formules du rang

C.1 Taille de la première matrice de rang déficient

Cette annexe a pour objectif de démontrer que la taille de la première matrice de rang déficient est :

$$n_a = \alpha_{min} \cdot n$$

où :

$$\alpha_{min} = \left\lfloor \frac{\mu^{\perp}}{n-k} + 1 \right\rfloor$$

Preuve :

Soit $\alpha_{min} = \lfloor a+1 \rfloor = a + \epsilon$ avec $\epsilon \in]0,\, 1]$.

$$\alpha_{min} = \left\lfloor \frac{\mu^{\perp}}{n-k} + 1 \right\rfloor = \frac{\mu^{\perp}}{n-k} + \epsilon \tag{C.1}$$

Il suffit de vérifier que :

$$\alpha_{min} \cdot n > \mu^{\perp} + \alpha_{min} \cdot k \geq (\alpha_{min} - 1) \cdot n \tag{C.2}$$

$$\begin{aligned}
\alpha_{min} \cdot n &= \left\lfloor \frac{\mu^{\perp}}{n-k} + 1 \right\rfloor \cdot n \\
&= \left(\frac{\mu^{\perp}}{n-k} + \epsilon \right) \cdot n \\
&= \frac{n \cdot \mu^{\perp}}{n-k} + \epsilon \cdot n \\
&= \frac{((n-k)+k) \cdot \mu^{\perp}}{n-k} + \epsilon \cdot n \\
&= \mu^{\perp} + \frac{k \cdot \mu^{\perp}}{n-k} + \epsilon \cdot n \\
&= \mu^{\perp} + \frac{k \cdot \mu^{\perp}}{n-k} + \epsilon \cdot k + \epsilon \cdot (n-k) \\
&= \mu^{\perp} + \left(\frac{\mu^{\perp}}{n-k} + \epsilon \right) \cdot k + \epsilon \cdot (n-k) \\
&= \mu^{\perp} + \left\lfloor \frac{\mu^{\perp}}{n-k} + 1 \right\rfloor \cdot k + \epsilon \cdot (n-k) \\
&= \mu^{\perp} + \alpha_{min} \cdot k + \epsilon \cdot (n-k)
\end{aligned} \tag{C.3}$$

Puisque $0 < \epsilon \leq 1$ et $0 < n - k \leq n$, on a $0 < \epsilon \cdot (n-k) \leq n$. D'où $\alpha_{min} \cdot n > \mu^{\perp} + \alpha_{min} \cdot k \geq (\alpha_{min} - 1) \cdot n$.

C.2 Symétrie entre les blocs $a_{l,i}$ et les blocs $b_{l,i}$

En utilisant le raisonnement par récurrence, nous démontrons ici que :

$$a_{l,i} = b_{l,\beta-1-i},\ \forall i \in [\![0, \beta-1]\!] \tag{C.4}$$

Preuve :

- Pour $i = 0$, on a $a_{l,0} = 0$ et $b_{l,\beta-1} = 0$. Alors, $a_{l,0} = b_{l,\beta-1}$. Par conséquent, l'équation (C.4) est vérifiée pour $i = 0$.
- On suppose que $a_{l,i} = b_{l,\beta-1-i}$ et on démontre que :

$$a_{l,i+1} = b_{l,\beta-1-(i+1)}$$

En utilisant les équations (3.16) et (3.17), $a_{l,i+1}$ est donné par :

$$\begin{aligned} a_{l,i+1} &= n - b_{l,i} \\ &= n - \mathrm{mod}(r - a_{l,i}, n) \\ &= \begin{cases} n - r + a_{l,i} & \text{si } a_{l,i} \leq r \\ a_{l,i} - r & \text{si } a_{l,i} > r \end{cases} \end{aligned}$$

– Si $a_{l,i} \leq r$, alors :

$$\begin{aligned} a_{l,i+1} &= n - r + a_{l,i} \\ &= n - r + b_{l,\beta-1-i} \\ &= n - r + \mathrm{mod}(r - a_{\beta-1-i}, n) \\ &= \begin{cases} n - a_{l,\beta-1-i} & \text{si } a_{l,\beta-1-i} \leq r \\ 2 \cdot n - a_{l,\beta-1-i} & \text{si } a_{l,\beta-1-i} > r \end{cases} \\ &= \begin{cases} n - n + b_{l,\beta-1-(i+1)} & \text{si } a_{l,\beta-1-i} \leq r \\ 2 \cdot n - n + b_{l,\beta-1-(i+1)} & \text{si } a_{l,\beta-1-i} > r \end{cases} \\ &= \begin{cases} b_{l,\beta-1-(i+1)} & \text{si } a_{l,\beta-1-i} \leq r \\ n + b_{l,\beta-1-(i+1)} & \text{si } a_{l,\beta-1-i} > r \end{cases} \end{aligned}$$

D'où, $a_{l,i+1} = b_{l,\beta-1-(i+1)}$ dans le cas de $a_{l,i} \leq r$ car $a_{l,i}$, $\forall i \in [\![0, \beta-1]\!]$, est strictement inférieur à n. Il reste à vérifier cette équation dans le cas de $a_{l,i} > r$.

– Si $a_{l,i} > r$, alors :

$$\begin{aligned} a_{l,i+1} &= a_{l,i} - r \\ &= b_{l,\beta-1-i} - r \\ &= \mathrm{mod}(r - a_{\beta-1-i}, n) - r \\ &= \begin{cases} r - a_{l,\beta-1-i} - r & \text{si } a_{l,\beta-1-i} \leq r \\ n + r - a_{l,\beta-1-i} - r & \text{si } a_{l,\beta-1-i} > r \end{cases} \\ &= \begin{cases} -n + b_{l,\beta-1-(i+1)} & \text{si } a_{l,\beta-1-i} \leq r \\ b_{l,\beta-1-(i+1)} & \text{si } a_{l,\beta-1-i} > r \end{cases} \end{aligned}$$

D'où, $a_{l,i+1} = b_{l,\beta-1-(i+1)}$ dans le cas de $a_{l,i} > r$ car $b_{l,i}$, $\forall i \in [\![0, \beta-1]\!]$, est strictement inférieur à n.

Par conséquent, l'équation (C.4) est vérifiée pour $i+1$.

C.3 Détermination de l'expression simplifiée de $a_{l,i}$

Cette annexe a pour objectif de simplifier l'expression de $a_{l,i}$ donnée dans l'équation (3.16). Pour traiter ce problème, nous procédons en trois étapes :

1. Démontrer que $a_{l,i}$ peut s'écrire :

$$a_{l,i} = \mathrm{mod}(i \cdot (n-r), n) \tag{C.5}$$

2. Vérifier qu'il existe $j \in [\![0, \beta-1]\!]$ qui satisfait $a_{l,i} = a_{l',j}$ pour tout $i \in [\![0, \beta-1]\!]$ tel que $\mathrm{pgcd}(l,n) = \mathrm{pgcd}(l',n) = d$ et $r' > r$.

3. Pour $r = (\beta-1) \cdot d$, démontrer que l'équation (C.5) s'écrit :

$$a_{l,i} = i \cdot d \tag{C.6}$$

Preuves :

1. Nous montrons par récurrence l'expression (C.5) :

   - Pour $i = 0$, on peut vérifier en utilisant l'équation (C.5) que $a_{l,0} = 0$.
   - Supposons que l'équation (C.5) est vraie pour l'étape i, on peut vérifier qu'elle est aussi vraie pour l'étape $i+1$, c'est-à-dire que :

$$a_{l,i+1} = \mathrm{mod}((i+1) \cdot (n-r), n) \tag{C.7}$$

   On a :

$$\begin{aligned} a_{l,i} &= \mathrm{mod}(i \cdot (n-r), n) \\ &= \begin{cases} i \cdot (n-r) & \text{si } i \cdot (n-r) < n \\ i \cdot (n-r) - q \cdot n & \text{si } i \cdot (n-r) \geq n \end{cases} \end{aligned} \tag{C.8}$$

   – Si $i \cdot (n-r) < n$ alors $a_{l,i} = i \cdot (n-r)$ avec $n > r$. En utilisant les équations (3.16) et (3.17), $a_{l,i+1}$ est donné par :

$$\begin{aligned} a_{l,i+1} &= n - b_{l,i} \\ &= n - \mathrm{mod}(r - i \cdot (n-r), n) \\ &= \begin{cases} n - r + i \cdot (n-r) & \text{si } i \cdot (n-r) \leq r \\ i \cdot (n-r) - r & \text{si } r < i \cdot (n-r) < n \end{cases} \\ &= \begin{cases} (i+1) \cdot (n-r) & \text{si } i \cdot (n-r) + n - r \leq r + n - r \\ i \cdot (n-r) - r + n - n & \text{si } r + n - r < i \cdot (n-r) + n - r < n + n - r \end{cases} \\ &= \begin{cases} (i+1) \cdot (n-r) & \text{si } (i+1) \cdot (n-r) \leq n \\ (i+1) \cdot (n-r) - n & \text{si } n < (i+1) \cdot (n-r) < 2 \cdot n - r \end{cases} \\ &= \mathrm{mod}((i+1) \cdot (n-r), n) \end{aligned} \tag{C.9}$$

   – Si $i \cdot (n-r) \geq n > r$ alors $a_{l,i} = i \cdot (n-r) - q \cdot n$ avec $n > r$ et $q = \left\lfloor \dfrac{i \cdot (n-r)}{n} \right\rfloor$. En

utilisant les équations (3.16) et (3.17), $a_{l,i+1}$ est donné par :

$$\begin{aligned} a_{l,i+1} &= n - b_{l,i} \\ &= n - \mathrm{mod}(r - i \cdot (n-r) - q \cdot n, n) \\ &= n - \mathrm{mod}(r - i \cdot (n-r), n) \\ &= n - (n + r - i \cdot (n-r)) \\ &= i \cdot (n-r) - r + n - n \\ &= (i+1) \cdot (n-r) - n \end{aligned} \tag{C.10}$$

Dans ce cas, on a $i \cdot (n-r) \geq n > r$, alors $i \cdot (n-r) + n - r > r + n - r \Rightarrow (i+1) \cdot (n-r) > n$.

D'après les équations (C.9) et (C.10), on peut déduire que l'équation (C.5) est vraie à l'étape $i+1$.

2. Soit $l = \gamma \cdot n + r$ et $l' = \delta \cdot n + r'$ tel que $r' > r$ et $\mathrm{pgcd}(r', n) = \mathrm{pgcd}(r, n) = d$, notre objectif est de démontrer que pour $i \in [\![0, \beta - 1]\!]$, il existe $j \in [\![0, \beta - 1]\!]$ tel que $a_{l,i} = a_{l',j}$. Ce problème sera traité par le raisonnement analyse-synthèse.

   - **Analyse :** Supposons qu'il existe j qui s'écrit :

$$j = \begin{cases} \dfrac{a_{l,i} + q \cdot n}{n - r'} & i \in]\!]0, \beta - 1]\!] \\ 0 & i = 0 \end{cases} \tag{C.11}$$

   avec $q \in [\![0, \dfrac{n - r'}{d}]\!]$ vérifiant que $a_{l,i} + q \cdot n$ divise $n - r'$.

   - **Synthèse :** On vérifie si cette expression de j peut satisfaire l'équation $a_{l,i} = a_{l',j}$. En utilisant l'équation (C.5), $a_{l',j}$ est donné par :

$$\begin{aligned} a_{l',j} &= \mathrm{mod}(j \cdot (n - r'), n) \\ &= \mathrm{mod}(\frac{a_{l,i} + q \cdot n}{n - r'} \cdot (n - r'), n) \\ &= \mathrm{mod}(a_{l,i} + q \cdot n, n) \\ &= \mathrm{mod}(a_{l,i}, n) \\ &= a_{l,i} \end{aligned} \tag{C.12}$$

3. Pour $r = (\beta - 1) \cdot d$, on démontre par le raisonnement par récurrence la relation (C.6) :

   - Si $i = 0$, on a $a_{l,0} = 0 = 0 \cdot d$. Alors la relation (C.6) est vérifiée.
   - On suppose que $a_{l,i} = i \cdot d$ et on démontre que :

$$a_{l,i+1} = (i+1) \cdot d$$

   Puisque $i \leq \beta - 1$ et $d > 0$, alors $i \cdot d \leq (\beta - 1) \cdot d \Rightarrow a_{l,i} \leq r$. En utilisant les relations (3.16) et (3.17), $a_{l,i+1}$ est obtenu par :

$$\begin{aligned} a_{l,i+1} &= n - b_{l,i} \\ &= n - \mathrm{mod}(r - a_{l,i}, n) \\ &= n - r + a_{l,i} \\ &= n - (\beta - 1) \cdot d + i \cdot d \\ &= n - \frac{n}{d} \cdot d + d + i \cdot d \\ &= (i+1) \cdot d \end{aligned} \tag{C.13}$$

C.4 Formule du rang dominant à partir de la formule générale

La formule générale $\varphi(l)$ permettant de calculer les déficiences de rang est :

$$\varphi(l) = k \cdot \left(\frac{l-n}{d} + 1\right) + \beta \cdot \mu^{\perp} + d \cdot \left\lfloor \frac{k}{d} \right\rfloor \cdot \left(\left\lfloor \frac{k}{d} \right\rfloor + 1\right) + 2 \cdot k \cdot \max\left\{0, \beta - 1 - \left\lfloor \frac{k}{d} \right\rfloor\right\} \tag{C.14}$$

Pour $l = \alpha \cdot n$, on démontre que cette formule est égale à :

$$\varphi(\alpha \cdot n) = \frac{k}{n} \cdot l + \mu^{\perp} \tag{C.15}$$

Preuve :

On a $l = \alpha \cdot n$, alors $d = \text{pgcd}(l, n) = n \Rightarrow \beta = 1$, donc :

$$\varphi(\alpha \cdot n) = k \cdot \left(\frac{n}{n} \cdot (\alpha - 1) + 1\right) + \mu^{\perp} + n \cdot \left\lfloor \frac{k}{n} \right\rfloor \cdot \left(\left\lfloor \frac{k}{n} \right\rfloor + 1\right) + 2 \cdot k \cdot \max\left\{0, -\left\lfloor \frac{k}{d} \right\rfloor\right\} \tag{C.16}$$

Puisque $k < n$, alors $\left\lfloor \frac{k}{n} \right\rfloor = 0$. D'où :

$$\begin{aligned} \varphi(\alpha \cdot n) &= \alpha \cdot k + \mu^{\perp} \\ &= \frac{k}{n} \cdot l + \mu^{\perp} \end{aligned} \tag{C.17}$$

ANNEXE D

Algorithme d'élimination de Gauss dans GF(q)

Cette annexe présente l'algorithme d'élimination de Gauss écrit dans le cas des corps de Galois. Cet algorithme transforme une matrice $\mathbf{R}_l$ en une matrice $\mathbf{T}_l$ et donne également en sortie une matrice $\mathbf{A}_l$ qui décrit les combinaisons des colonnes de $\mathbf{R}_l$.

1. Initialiser la matrice $\mathbf{T}_l$ avec $\mathbf{R}_l$ et la matrice $\mathbf{A}_l$ avec la matrice identité $\mathbf{I}_l$, puis commencer par la première colonne $\mathbf{t}_i^{(l)}$, avec $i = 1$;

2. Si $\mathbf{t}_i^{(l)}$ n'est pas une colonne nulle (tous ses éléments sont nuls), prendre le premier élément non nul de cette colonne situé à la j-ème ligne, $t_i^{(l)}(j)$, et chercher un nombre non-nul $v \in \mathrm{GF}(2^m)$ tel que $t_i^{(l)}(j) \cdot v = 1$;

3. Multiplier $\mathbf{t}_i^{(l)}$ et $\mathbf{a}_i^{(l)}$ par v ;

4. Effectuer des combinaisons de colonnes afin de rendre tous les coefficients de la même ligne dans la i'-ème colonne nuls, où $i' > i$. Appliquer les mêmes opérations à la matrice $\mathbf{A}_l$,

5. Réitérer cette procédure avec la colonne $i + 1$ et revenir à l'étape 2.

ANNEXE

E Calcul détaillé de la probabilité P_i

La probabilité P_i d'avoir un coefficient nul dans la i-ème colonne de la matrice $\tilde{\mathbf{T}}(l)_i$ est initialement exprimée par l'équation :

$$P_i = \frac{1 + (q-1) \cdot \left(1 - \frac{p_e \cdot q}{q-1}\right)^{N_i(l)}}{q} \tag{E.1}$$

Notons $P_1(s) = Pr\left[X = s\right]$ et $P_2(s) = Pr\left[\sum_{k=1}^{s} a_i^{(l)}(k) \cdot e_k^{(l)}(j) = 0\right]$. Alors, la probabilité P_i sera égale à :

$$P_i = P_1(0) + \sum_{s=1}^{N_i(l)} P_1(s) \cdot P_2(s) \tag{E.2}$$

Calcul de la probabilité $P_1(s)$

Nous faisons l'hypothèse que les erreurs sont indépendantes les unes des autres et uniformément distribuées dans GF$(q)^*$. La variable X suit une loi binomiale de paramètres $N_i(l)$ et p_e. Alors, la probabilité $P_1(s)$ peut s'écrire :

$$P_1(s) = \binom{N_i(l)}{s} \cdot p_e^s \cdot (1-p_e)^{N_i(l)-s} \tag{E.3}$$

avec p_e la probabilité d'erreur d'un symbole non-binaire.

Calcul de la probabilité $P_2(s)$

La probabilité $P_2(s)$ correspond à la probabilité d'avoir $\sum_{k=1}^{s} a_i^{(l)}(k) \cdot e_k^{(l)}(j) = 0$ avec $e_k^{(l)}(j) \in$GF$(q)^*$. Nous notons $\overline{P_2(s)}$ la probabilité d'avoir la somme $\sum_{k=1}^{s} a_i^{(l)}(k) \cdot e_k^{(l)}(j) \neq 0$. Cette probabilité est donnée par :

$$\overline{P_2(s)} = 1 - P_2(s) \tag{E.4}$$

Nous démontrons par récurrence que la probabilité $P_2(s)$ peut être déterminée par :

$$P_2(s) = \frac{1 - P_2(s-1)}{q-1} \tag{E.5}$$

On a $P_2(0) = 1$ car il n'y a pas de positions erronées. Dans le cas d'une seule position erronée, on a $P_2(1) = 0$. Par contre, en prenant l'exemple du corps GF(2^2), la probabilité $P_2(s = 2)$ est déterminée par la matrice M dont les indices de lignes et de colonnes correspondent aux éléments

non-nuls de ce corps. Les coefficients de cette matrice correspondent à la somme dans GF(2^2) des indices de ligne et de colonne.

$$M = \begin{pmatrix} 0 & 3 & 2 \\ 3 & 0 & 1 \\ 2 & 1 & 0 \end{pmatrix} \tag{E.6}$$

Si on a $(a_i^{(l)}(1), e_1^{(l)}(j)) \in (GF(2^2)^*)^2$ et $(a_i^{(l)}(2), e_2^{(l)}(j)) \in (GF(2^2)^*)^2$, alors la probabilité d'avoir $a_i^{(l)}(1) \cdot e_1^{(l)}(j) + a_i^{(l)}(2) \cdot e_2^{(l)}(j) = 0$ est $P_2(2) = 3/9 = 1/3$. La probabilité calculée vérifie bien la formule (E.5).

Nous supposons que la formule (E.5) est vérifiée pour s et nous la démontrons pour $s+1$. Si on a $\sum_{k=1}^{s+1} a_i^{(l)}(k) \cdot e_k^{(l)}(j) = 0$ alors on aura $e_{s+1}^{(l)} = -\frac{1}{a_i^{(l)}(s+1)} \cdot \sum_{k=1}^{s} a_i^{(l)}(k) \cdot e_k^{(l)}(j)$ qui appartient au corps GF$(q)^*$ avec une probabilité $1/(q-1)$. Par conséquent, la probabilité $P_2(s+1)$ est déterminée par :

$$\begin{aligned} P_2(s+1) &= Pr\left(e_{s+1}^{(l)} \in GF(q)^*, \textstyle\sum_{k=1}^{s} a_i^{(l)}(k) \cdot e_k^{(l)}(j) \neq 0\right) \\ &= Pr\left(e_{s+1}^{(l)} \in GF(q)^*\right) \cdot Pr\left(\textstyle\sum_{k=1}^{s} a_i^{(l)}(k) \cdot e_k^{(l)}(j) \neq 0\right) \\ &= \frac{P_2(s)}{q-1} \\ &= \frac{1 - P_2(s)}{q-1} \end{aligned} \tag{E.7}$$

Notre objectif, par la suite, est de simplifier l'expression de $P_2(s)$. En effet, un changement de variable peut être effectué en considérant $\varphi(s) = (q-1)^{s-1} \cdot P_2(s)$. En remplaçant $P_2(s)$ par $\varphi(s)$, l'expression (E.5) devient :

$$\begin{aligned} \frac{\varphi(s)}{(q-1)^{s-1}} &= \frac{1 - \frac{\varphi(s-1)}{(q-1)^{s-2}}}{q-1} \\ \frac{\varphi(s)}{(q-1)^{s-2}} &= \frac{(q-1)^{s-2} - \varphi(s-1)}{(q-1)^{s-2}} \\ \varphi(s) + \varphi(s-1) &= (q-1)^{s-2} \end{aligned} \tag{E.8}$$

Si $\rho(s) = (-1)^s \cdot \varphi(s)$, l'expression (E.8) peut s'écrire :

$$\begin{aligned} (-1)^s \cdot \rho(s) + (-1)^{s-1} \cdot \rho(s-1) &= (q-1)^{s-2} \\ -\rho(s) + \rho(s-1) &= (-1)^{s-1} \cdot (q-1)^{s-2} \\ \rho(s) &= \rho(s-1) + (1-q)^{s-2} \\ \rho(s) &= \rho(s-2) + (1-q)^{s-2} + (1-q)^{s-3} \\ \rho(s) &= \rho(1) + \textstyle\sum_{i=0}^{s-1} (1-q)^i \end{aligned} \tag{E.9}$$

Or, $\sum_{i=0}^{s-1} (1-q)^i$ est la somme d'une suite géométrique de raison $1-q$. Alors, elle peut s'écrire :

$$\begin{aligned} \textstyle\sum_{i=0}^{s-1} (1-q)^i &= \frac{(1-q)^{s-1} - 1}{(1-q) - 1} \\ &= \frac{1 - (-1)^{s-1} \cdot (q-1)^{s-1}}{q} \end{aligned}$$

La variable $\rho(1)$ est calculée par :

$$\begin{aligned}\rho(1) &= -\varphi(1)\\ &= -(q-1)^{1-1}\cdot P_2(1)\\ &= -\frac{1-P_2(0)}{q-1}\\ &= 0\end{aligned}$$

D'où, $\rho(s)$ sera égale à :

$$\rho(s) = \frac{1-(-1)^{s-1}\cdot(q-1)^{s-1}}{q} \tag{E.10}$$

Par conséquent, l'expression simplifiée de $P_2(s)$ est donnée par :

$$P_2(s) = \frac{(-1)^s + (q-1)^{s-1}}{q\cdot(q-1)^{s-1}} \tag{E.11}$$

Calcul de la probabilité globale P_i

En utilisant les deux expressions (E.3) et (E.11), la probabilité globale P_i est donnée par :

$$\begin{aligned}P_i &= \textstyle\sum_{s=0}^{N_i(l)} \binom{N_i(l)}{s}\cdot p_e^s\cdot(1-p_e)^{N_i(l)-s}\cdot\frac{(-1)^s+(q-1)^{s-1}}{q\cdot(q-1)^{s-1}}\\ q\cdot P &= \textstyle\sum_{s=0}^{N_i(l)} \binom{N_i(l)}{s}\cdot p_e^s\cdot(1-p_e)^{N_i(l)-s}\cdot\left(1+\left(\frac{-1}{q-1}\right)^s\cdot(q-1)\right)\\ &= \textstyle\sum_{s=0}^{N_i(l)} \binom{N_i(l)}{s}\cdot p_e^s.(1-p_e)^{N_i(l)-s} + (q-1)\cdot\sum_{s=0}^{N_i(l)}\binom{N_i(l)}{s}\cdot\left(\frac{-p_e}{q-1}\right)^s\cdot(1-p_e)^{N_i(l)-s}\end{aligned} \tag{E.12}$$

Or, on sait que :

$$(Z+Y)^{N_i(l)} = \sum_{s=0}^{N_i(l)}\binom{N_i(l)}{s}\cdot Z^s.Y^{N_i(l)-s}$$

Alors, la formule (E.12) devient :

$$\begin{aligned}q.P &= (p_e+1-p_e)^{N_i(l)} + (q-1)\cdot\left(\frac{-p_e}{q-1}+1-p_e\right)^{N_i(l)}\\ &= 1+(q-1)\cdot\left(1-\frac{-p_e}{q-1}\cdot(1+q-1)\right)^{N_i(l)}\end{aligned} \tag{E.13}$$

Par conséquent, la probabilité qu'un élément de la i-ème colonne de la matrice $\tilde{\mathbf{T}}_l$ soit égal égale à 0 est déterminée par l'équation (E.1).

ANNEXE

F Calcul du seuil optimal $\hat{\eta}_{opt}$

L'objectif de cette annexe est de déterminer le seuil optimal $\hat{\eta}_{opt}$ qui annule la dérivée première par rapport à $\hat{\eta}$ de la probabilité de mauvaise détection P_{wd} donnée par :

$$P_{wd} = 1 - \phi\left(\frac{\hat{\eta} - \mu_1}{\sigma_1}\right) + \phi\left(\frac{\hat{\eta} - \mu_0}{\sigma_0}\right) \tag{F.1}$$

où la fonction $\phi(x)$ est définie par l'équation (4.29).

Calcul de la dérivée :

La dérivée première de P_{wd} peut s'écrire :

$$\frac{\partial P_{wd}}{\partial \eta} = \frac{\partial \phi\left(\frac{\hat{\eta} - \mu_0}{\sigma_0}\right)}{\partial \eta} - \frac{\partial \phi\left(\frac{\hat{\eta} - \mu_1}{\sigma_1}\right)}{\partial \eta} \tag{F.2}$$

où :

$$\begin{cases} \dfrac{\partial \phi\left(\frac{\hat{\eta} - \mu_0}{\sigma_0}\right)}{\partial \eta} = \dfrac{1}{\sigma_0\sqrt{2\cdot\pi}} \cdot e^{-\frac{(\hat{\eta} - \mu_0)^2}{2\sigma_0^2}} \\ \dfrac{\partial \phi\left(\frac{\hat{\eta} - \mu_1}{\sigma_1}\right)}{\partial \eta} = \dfrac{1}{\sigma_1\sqrt{2\cdot\pi}} \cdot e^{-\frac{(\hat{\eta} - \mu_1)^2}{2\sigma_1^2}} \end{cases} \tag{F.3}$$

Résolution de l'équation $\dfrac{\partial P_{wd}}{\partial \eta} = 0$:

En utilisant l'équation (F.3), l'expression $\dfrac{\partial P_{wd}}{\partial \eta} = 0$ est exprimée par :

$$\frac{\partial P_{wd}}{\partial \eta} = \frac{1}{\sigma_0\sqrt{2\cdot\pi}} \cdot e^{-\frac{(\hat{\eta} - \mu_0)^2}{2\sigma_0^2}} - \frac{1}{\sigma_1\sqrt{2\cdot\pi}} \cdot e^{-\frac{(\hat{\eta} - \mu_1)^2}{2\sigma_1^2}} = 0 \tag{F.4}$$

L'équation (F.4) devient :

$$\sigma_1 \cdot e^{-\frac{(\hat{\eta} - \mu_0)^2}{2\sigma_0^2}} = \sigma_0 \cdot e^{-\frac{(\hat{\eta} - \mu_1)^2}{2\sigma_1^2}} \tag{F.5}$$

Lorsqu'on applique le logarithme népérien (ln), l'équation (F.5) devient :

$$\frac{(\hat{\eta}-\mu_0)^2}{2\sigma_0^2} - \frac{(\hat{\eta}-\mu_1)^2}{2\sigma_1^2} - \ln\left(\frac{\sigma_1}{\sigma_0}\right) = 0 \tag{F.6}$$

Le développement de l'équation (F.6) conduit à :

$$\hat{\eta}^2 \cdot \left(\sigma_1^2 - \sigma_0^2\right) + 2 \cdot \hat{\eta} \cdot \left(\sigma_0^2 \cdot \mu_1 - \sigma_1^2 \cdot \mu_0\right) - \sigma_0^2 \cdot \mu_1^2 + \sigma_1^2 \mu_0^2 - 2 \cdot \sigma_0^2 \cdot \sigma_1^2 \cdot \ln\left(\frac{\sigma_1}{\sigma_0}\right) = 0 \tag{F.7}$$

Avec les notations suivantes :

$$\begin{cases} a = \sigma_1^2 - \sigma_0^2 \\ b = 2 \cdot \left(\mu_1 \cdot \sigma_0^2 - \mu_0 \cdot \sigma_1^2\right) \\ c = \mu_0^2 \cdot \sigma_1^2 - \mu_1^2 \cdot \sigma_0^2 - 2 \cdot \sigma_0 \cdot \sigma_1 \cdot \ln\left(\frac{\sigma_1}{\sigma_0}\right) \end{cases}$$

on obtient une équation de second degré donnée par l'équation :

$$a \cdot \hat{\eta}^2 + b \cdot \hat{\eta} + c = 0 \tag{F.8}$$

En remplaçant les paramètres μ_0, μ_1, σ_0^2 et σ_1^2 par leurs valeurs données par :

$$\begin{cases} \mu_0 = M \cdot P_i \\ \mu_1 = \dfrac{M}{q} \\ \sigma_0^2 = M \cdot P_i \cdot (1 - P_i) \\ \sigma_1^2 = \dfrac{M \cdot (q-1)}{q^2} \end{cases}$$

les coefficients de l'équation de second degré deviennent :

$$\begin{cases} a = \dfrac{M}{q^2} \cdot \left(q - 1 - q^2 \cdot P_i \cdot (1 - P_i)\right) \\ b = \dfrac{2 \cdot M^2}{q^2} \cdot (1 - P_i^2) \\ c = \dfrac{M^3}{q^3} \cdot P_i \cdot (q \cdot P_i - 1) - \dfrac{M}{q} \cdot \sqrt{P_i \cdot (1 - P_i) \cdot (q-1)} \cdot \ln\left(\dfrac{q-1}{q^2 \cdot (1 - P_i) \cdot P_i}\right) \end{cases}$$

Donc, le seuil optimal $\hat{\eta}_{opt}$ sera une solution de l'équation (F.8).

ANNEXE G

Calcul du seuil de décision τ

L'objectif de cette annexe est de démontrer que le seuil de décision τ peut être déterminé par l'équation :

$$\tau = 1 - \sigma \cdot \exp(1) \tag{G.1}$$

Lors de l'utilisation d'une modulation BPSK, un symbole r_i reçu en sortie d'un canal BBAG est donné par :

$$r_i = c_i + e_i = \pm 1 + e_i$$

où e_i est un bruit blanc additif gaussien centré de variance σ^2 et c_i est un symbole modulé transmis. Notons $\mathcal{H}_0$ et $\mathcal{H}_1$ deux hypothèses tel que :

$$\begin{cases} \text{Si } c_i = 1 & \text{alors } \mathcal{H}_0 \\ \text{Si } c_i = -1 & \text{alors } \mathcal{H}_1 \end{cases}$$

Les densités de probabilités conditionnelles du symbole reçu r_i sont données par :

$$f\left(r_i|\mathcal{H}_0\right) = \frac{1}{\sigma \cdot \sqrt{2 \cdot \pi}} \cdot e^{-\frac{1}{2 \cdot \sigma^2} \cdot (r_i - 1)^2} \tag{G.2}$$

$$f\left(r_i|\mathcal{H}_1\right) = \frac{1}{\sigma \cdot \sqrt{2 \cdot \pi}} \cdot e^{-\frac{1}{2 \cdot \sigma^2} \cdot (r_i + 1)^2} \tag{G.3}$$

Le rôle du démodulateur au niveau du récepteur est de décider si le symbole reçu r_i est un 1 ou -1 lors d'une décision ferme. Pour ce faire, nous fixons un seuil de décision noté τ. Alors, nous définissons deux hypothèses en fonction de τ :

$$\begin{cases} \text{Si} & r_i > \tau & \text{alors} & r_i = 1 \\ \text{Si} & r_i < -\tau & \text{alors} & r_i = -1 \end{cases} \tag{G.4}$$

Considérons qu'il existe une mauvaise détection du symbole -1. La probabilité de mauvaise détection de ce symbole défini par $Pr\left[r_i > -\tau | c_i = -1\right]$ correspond à :

$$\begin{aligned} Pr\left[r_i > -\tau | c_i = -1\right] &= \int_{-\tau}^{+\infty} f\left(r_i|\mathcal{H}_1\right) \cdot dr_i \\ &= \frac{1}{\sigma \cdot \sqrt{2 \cdot \pi}} \cdot \int_{-\tau}^{+\infty} e^{-\frac{1}{2 \cdot \sigma^2} \cdot (r_i + 1)^2} \cdot dr_i \\ &= \frac{1}{\sqrt{2 \cdot \pi}} \cdot \int_{\frac{-\tau+1}{\sigma}}^{+\infty} e^{\frac{-t^2}{2}} \cdot dt \\ &= 1 - \frac{1}{\sqrt{2 \cdot \pi}} \cdot \int_{-\infty}^{\frac{-\tau+1}{\sigma}} e^{\frac{-t^2}{2}} \cdot dt \end{aligned} \tag{G.5}$$

Nous considérons que la probabilité $Pr\left[r_i > -\tau | c_i = -1\right]$ est fixée à 0.0033. Alors, l'équation (G.5)

peut s'écrire :

$$\phi\left(\frac{-\tau+1}{\sigma}\right) = \frac{1}{\sqrt{2\cdot\pi}}\cdot\int_{-\infty}^{\frac{-\tau+1}{\sigma}} e^{\frac{-t^2}{2}}\cdot dt = 1 - 0.0033 = 0.9967 \tag{G.6}$$

En utilisant la table de fonction de répartition normale réduite, nous avons $\frac{-\tau+1}{\sigma} = 2.72 = \exp(1)$. Dans ce cas, le seuil τ peut être déterminé par l'équation (G.1).

ANNEXE

H Quelques fonctions universelles dans GF(2^3)

Cette annexe présente les résultats de transformation des éléments de GF(2^3), défini par le polynôme primitif $x^3 + x + 1$, par quelques fonctions universelles de mapping-demapping.

GF(2^3)		σ		m_α		$m_\alpha \circ \sigma$	
0	0	0	0	0	0	0	0
1	1	1	1	α	2	α	2
α	2	α^2	4	α^2	4	α^3	3
α^2	4	α^4	6	α^3	3	α^5	7
α^3	3	α^6	5	α^4	6	1	1
α^4	6	α	2	α^5	7	α^2	4
α^5	7	α^3	3	α^6	5	α^4	6
α^6	5	α^5	7	1	1	α^6	5

Bibliographie

[3GP05a] 3GPP TS 05.03 v8.9.0. *Channel coding (release 1999)*. The 3rd Generation Partnership Project, Technical Specification Group GSM/EDGE Radio Access Network, January 2005. http ://www.3gpp.org.

[3GP05b] 3GPP TS 25.212 v6.5.0. *Multiplexing and channel coding (FDD) (release 6)*. The 3rd Generation Partnership Project, Technical Specification Group Radio Access Network, June 2005. http ://www.3gpp.org.

[3GP09] 3GPP2 C.S0002-E v0.99. *Physical layer Standard for CDMA2000 Spread Spectrum Systems (Revision E)*. The 3rd Generation Partnership Prject 2, February 2009. http ://www.3gpp2.org.

[Bar05] J. Barbier. Reconstruction of turbo-code encoders. In *Proceedings of SPIE Security and Defence, Space Communication Technologies Symposium*, volume 5819, pages 463–473, Orlando, FL, USA, 2005.

[Bar07] Johann Barbier. *Analyse de canaux de communication dans un contexte non coopératif : application aux codes correcteurs d'erreurs et à la stéganalyse*. Thèse de doctorat, Palaiseau, Ecole polytechnique, janvier 2007.

[BCJR74] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv. Optimal decoding of linear codes for minimizing symbol error rate (corresp.). *IEEE Transactions on Information Theory*, 20(2) :284–287, 1974.

[BD03] L. Barnault and D.Declercq. Fast decoding algorithm for LDPC over GF(2^q). In *Proceedings ITW*, pages 70–73, Paris, France, Mar. 2003.

[Ber06] Claude Berrou. *Codes et turbocodes*. Springer Editions, 2006.

[BG03] G. Burel and R. Gautier. Blind estimation of encoder and interleaver characteristics in a non cooperative context. In *IASTED International Conference on Communications, Internet and Information Technology*, Scottsdale, AZ, USA, 2003.

[BL09] J. Barbier and J. Letessier. Forward error correcting codes characterization based on rank properties. In *Proc. of IEEE International Conference on Wireless Communications and Signal Processing, WCSP09,*, Nanjing, China, 2009.

[Bog05] Patrick Bogaert. *Probabilités pour scientifiques et ingénieurs : Introduction au calcul des probabilités*. De Boeck Supérieur, November 2005.

[BRC60] R. C. Bose and D. K. Ray-Chaudhuri. On a class of error correcting binary group codes. *Information and Control*, 3(3) :68–79, 1960.

[BS08] J.A Briffa and H.G Schaathun. Non-binary turbo codes and applications. In *5th International Symposium on Turbo Codes and Related Topics*, Lausanne, Sept. 2008.

[BSH06] J. Barbier, G. Sicot, and S. Houcke. Algebraic approach for the reconstruction of linear and convolutional error correcting codes. *International Journal of Applied Mathematics and Computer Sciences*, 2(3) :113–118, 2006.

[Can98a] A. Canteaut. A new algorithm for finding minimum-weight words in a linear code : Application to mcelieces cryptosystem and to narrow-sense bch codes of length 511. *IEEE Transactions on Information Theory*, 44 :367–378, 1998.

[Can98b] Anne Canteaut. A new algorithm for finding minimum-weight words in a linear code : Application to mcelieces cryptosystem and to narrow-sense bch codes of length 511. *IEEE Transactions on Information Theory*, 44 :367–378, 1998.

[CCW+07] W. R. Carson, I. Chatzigeorgiou, I. J. Wassell, M. R. D. Rodrigues, and R. Carrasco. On the performance of iterative demapping and decoding techniques over quasi-static fading channels. In *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 1–5, 2007.

[CDE+05] Jinghu Chen, A. Dholakia, E. Eleftheriou, M.P.C. Fossorier, and Xiao-Yu Hu. Reduced-complexity decoding of LDPC codes. *IEEE Transactions on Communications*, 53(8) :1288–1299, 2005.

[CF09] M. Cluzeau and M. Finiasz. Recovering a code's length and synchronization from a noisy intercepted bitstream. In *IEEE International Symposium on Information Theory, 2009. ISIT 2009*, pages 2737–2741, 2009.

[Clu06a] M. Cluzeau. Block code reconstruction using iterative decoding techniques. In *2006 IEEE International Symposium on Information Theory*, pages 2269–2273, 2006.

[Clu06b] M. Cluzeau. *Reconnaissance d'un schéma de codage.* Thèse de doctorat, Ecole Polytechnique, France, 2006.

[Cra91] J.W. Craig. A new, simple and exact result for calculating the probability of error for two-dimensional signal constellations. In *Military Communications Conference*, volume 2, pages 571–555, 1991.

[CSA95] J. B. Chennakeshu, Sandeep, and Anderson. Error rates for rayleigh fading multichannel reception of mpsk signals. *IEEE Transactions on Communications*, 43(234) :338–346, 1995.

[DA06] Ahmed H Desoky and Aleksey Y Ashikhmin. Cryptography software system using galois field arithmetic. In *IEEE Information Assurance Workshop*, pages 386–387, 2006.

[DCG04] D. Declercq, M. Colas, and G. Gelle. Regular gf(2^q)-ldpc coded modulations for higher order qam-awgn channels. In *ISITA 04*, Parme, Italy, 2004.

[DF05] D. Declercq and M. Fossorier. Extended minsum algorithm for decoding LDPC codes over GF(q). In *Proceedings ISIT*, page 464, Adelaide, Australia, Sept. 2005.

[DF07] D. Declercq and M. Fossorier. Decoding algorithms for nonbinary LDPC codes over GF(q). *IEEE Transactions on Communications*, 55(4) :633–643, 2007.

[DM98a] M. Davey and D. MacKay. Low density parity check codes over gf(q). In *Information Theory Workshop*, pages 70–71, 1998.

[DM98b] M.C. Davey and D. MacKay. Low-density parity-check codes over GF(q). *IEEE Communications Letters*, 2 :165–167, 1998.

[Eli54] P. Elias. Error-free coding. *Transactions of the IRE Professional Group on Information Theory*, 4(4) :29–37, 1954.

[Fil97] E. Filiol. Reconstruction of convolutional encoders over gf(p). In *Cryptography and Coding ; proceedings of the 6th IMA Conference, number 1355*, 1997.

[Fil00] E. Filiol. Reconstruction of punctured convolutional encoders. In *International Symposium on Information Theory and Application (ISITA)*, pages 4–7, Hawai, USA, November 2000.

[Fil01] E. Filiol. *Technique de reconstruction en cryptologie et théorie des codes.* Thèse de doctorat, École Polytechnique, France, 2001.

[For70] Jr. Forney, G.D. Convolutional codes i : Algebraic structure. *IEEE Transactions on Information Theory*, 16(6) :720–738, 1970.

[Gal62] R. G. Gallager. Low-density parity-check codes. *IRE Transactions on Information Theory*, 8(1) :21–28, 1962.

[Gal63] R. G. Gallager. *Low-Density Parity-Check Codes.* Phd thesis, MIT press, Cambridge, 1963.

[GKPP06] J. Guajardo, S. S. Kumar, C. Paar, and J. Pelzl. Efficient software-implementation of finite fields with applications to cryptography. *Acta Applicandae Mathematica*, 93(1-3) :3–32, September 2006.

[Gu05] Nong Gu. Adaptive blind channel equalization based on constellation information of mpsk signal. In *IEEE International Conference on Industrial Technology. ICIT 2005*, pages 146–151, 2005.

[Ham50] R. W. Hamming. Error detecting and error correcting codes. *BELL System Technical Journal*, 29(2) :147–160, 1950.

[HEA05] X. Y. Hu, E. Eleftheriou, and D.-M. Arnold. Regular and irregular progressive edge-growth tanner graphs. *IEEE Transactions on Information Theory*, 51(1) :386–398, 2005.

[Hoc59] A. Hocquenghem. Codes correcteurs d'erreurs. *Chiffres*, 2 :147–156, 1959.

[Hos10] M.T. Hossain. Cooperative communications : Synchronization in fast flat-fading channels with various signal constellations. In *IEEE Eleventh International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pages 1–5, 2010.

[Jab92] N.K. Jablon. Joint blind equalization, carrier recovery and timing recovery for high-order qam signal constellations. *IEEE Transactions on Signal Processing*, 40(6) :1383–1398, June 1992.

[JW01] S.J. Johnson and Steven R. Weller. Construction of low-density parity-check codes from kirkman triple systems. In *IEEE Global Telecommunications Conference, GLOBECOM '01*, volume 2, pages 970–974, 2001.

[KCA11] M. Karabacak, H. A. Cirpan, and Hüseyin Arslan. Cooperative communications : Synchronization in fast flat-fading channels with various signal constellations. In *IEEE Radio and Wireless Symposium*, pages 146–149, 2011.

[KLF01] Y. Kou, S. Lin, and P. C. Fossorier. Low density parity check codes based on finite geometries : A rediscovery and new results. *IEEE Trans. Inform. Theory*, 47 :2711–2736, 2001.

[LTC98] J. Lu, T. T. Tjhung, and C. C. Chai. Error probability performance of l-branch diversity reception of mqam in rayleigh fading. *IEEE Transactions on Communications*, 46(2) :179–181, 1998.

[Mac99] D.J.C. MacKay. Good error-correcting codes based on very sparse matrices. *IEEE Trans. Inform. Theory*, 45(2) :399–431, 1999.

[Mar09] M. Marazin. *Reconnaissance en aveugle de codeur à base de code convolutif : Contribution à la mise en oeuvre d'un récepteur intelligent.* Thèse de doctorat, Université de Bretagne Occidentale - Brest, December 2009.

[MD99] D.J.C. MacKay and M.C. Davey. Evaluation of gallager codes for short block length and high rate applications. In *In Codes, Systems and Graphical Models*, pages 113–130, 1999.

[Men93] A.J. Menezes. *Applications of Finite Fields.* Kluwer Academic Publishers, 1993.

[MGB09a] M. Marazin, R. Gautier, and G. Burel. Blind recovery of the second convolutional encoder of a turbo-code when its systematic outputs are punctured. *MTA*, XIX(2) :213–232, 2009.

[MGB09b] M. Marazin, R. Gautier, and G. Burel. Dual code method for blind identification of convolutional encoder for cognitive radio receiver design. In *IEEE GLOBECOM Workshops*, 2009.

[MGB11] M. Marazin, R. Gautier, and G. Burel. Blind recovery of k/n rate convolutional encoders in a noisy environment. *EURASIP Journal on Wireless Communications and Networking*, 2011(168) :1–9, 2011.

[MN96] D.J.C. MacKay and R.M. Neal. Near shannon limit performance of low density parity check codes. *Electron. Lett.*, 32(18) :1645–1646, 1996.

[Mor12] E. Martinez Moro. *Algebraic Geometry Modeling in Information Theory.* World Scientific, 2012.

[Pla96] G. Planquette. *Identification de trains binaires codés.* Thèse de doctorat, Université de Rennes I, December 1996.

[Pre92] O. Pretzel. *Error-correcting codes and finite fields.* Clarendon Press, 1992.

[Pro01] John G. M Proakis. *Digital communications.* McGraw-Hill, Boston, 2001.

[Rao81] T. R. N. Rao. Arithmetic of finite fields. In *IEEE 5th Symposium on Computer Arithmetic (ARITH)*, pages 146–149, Ann Arbor, MI, USA, 1981.

[Ree54] I. Reed. A class of multiple-error-correcting codes and the decoding scheme. *IEEE Transactions on Information Theory*, 4(4) :38–49, 1954.

[Ric95] B. Rice. Determining parameters of a rate $1/n$ convolutional encoder over gf(q). In *Proceedings of 3-rd International Conference on Finite Fields and Applications*, Glasgow, 1995.

[Ric03] T. Richardson. Error floors of ldpc codes. In *Proc. 41st Annual Allerton Conf on Communications Control and Computing*, 2003.

[RL09] William R. and Shu L. *Channel Codes : Classical and Modern.* Cambridge University Press, September 2009.

[RS] I. S. Reed and G. Solomon. Polynomial codes over certain finite fields.

[RS60] I. Reed and G. Solomon. Polynomial codes over certain finite fields. *Journal of the Society of Industrial and Applied Mathematics*, 8(2) :300–304, 1960.

[RSU01] T.J. Richardson, M.A. Shokrollahi, and R.L. Urbanke. Design of capacity-approaching irregular low-density parity-check codes. 47(2) :619–637, 2001.

[RW87] W. E. Ryan and S. G. Wilson. Convolutional codes over GF(q) with applications to frequency-hopping channels. In *MILCOM*, page 72, October 1987.

[RW91] W. E. Ryan and S. G. Wilson. Two classes of convolutional codes over GF(q) for q-ary orthogonalsignaling. *IEEE Transactions on Communications*, 39 :30, 1991.

[Rya91] W.E. Ryan. Two classes of convolutional codes over gf(q) for q -ary orthogonal signaling. *IEEE Transactions on Communications*, 39(1) :30–40, Jan 1991.

[Sat75] Y. Sato. A method of self-recovering equalization for multilevel amplitude-modulation systems. *IEEE Transactions on Communications*, 23(6) :679–682, June 1975.

[SBW94] V. K. Bhargava Stephen B. Wicker. *Reed-Solomon Codes and Their Applications.* John Wiley and Sons Inc, 1994.

[SF02] D. Sridhara and T.E. Fuja. Low density parity check codes defined over groups and rings. In *proc. of Information Theory Workshop (ITW)*, 2002.

[SH05] G. Sicot and S. Houcke. Blind detection of interleaver parameters. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 3, pages 829–832, Philadelphia, Pennsylvania, 2005.

[Sha48] C. E. Shannon. A mathematical theory of communication. *Bell system technical journal*, 27 :379–423, 623–656, July, October 1948.

[Sha10] B. Shams. *Les Codes LDPC non-binaires de nouvelle génération.* Cergy Pontoise, December 2010.

[Sha11] Soleymani M.R Shayegh, F. Efficient iterative techniques for soft decision decoding of reed-solomon codes. *IEEE Transactions on Communications*, 59(2) :428–436, 2011.

[SHB09] G. Sicot, S. Houcke, and J. Barbier. Blind detection of interleaver parameters. *Signal Processing*, 89(4) :450–462, April 2009.

[SK10] E. Savas¸ and Ç. Kaya Koç. Finite field arithmetic for cryptography. *IEEE Circuits and Systems Magazine*, 10(2) :40–56, 2010.

[Ste89a] J. Stern. A method for finding codewords of small weight. In *Coding Theory and Applications*, number 388, pages 106–113. Springer Berlin Heidelberg, January 1989.

[Ste89b] J. Stern. A method for finding codewords of small weight. In *Coding Theory and Applications*, number 388, pages 106–113. Springer Berlin Heidelberg, January 1989.

[Tan81] R. M. Tanner. A recursive approach to low complexity codes. *IEEE Transactions on Information Theory*, pages 533–547, 1981.

[Tod05] *Error Correction Coding : Mathematical Methods and Algorithms.* Wiley-Interscience, 2005.

[VA05] P. K. Vittahaladevuni and M.-S. Alouini. Effect of imperfect phase and timing synchronization on the bit-error rate performance of psk modulations. *IEEE Transactions on Communications*, 53(7) :1096–1099, July 2005.

[Val00] A. Valambois. *Décodage, détection et reconnaissance des codes linéaires binaires*. Thèse de doctorat, Université de Limoges, France, Octobre 2000.

[Val01] A. Valembois. Detection and recognition of a binary linear code. *Discrete Applied Mathematics*, 111(1-2) :199–218, 2001.

[Vas01] B. Vasic. Structured iteratively decodable codes based on steiner systems and their application in magnetic recording. In *IEEE Global Telecommunications Conference, 2001. GLOBECOM '01*, volume 5, pages 2954–2960, 2001.

[Vas02] B. Vasic. Combinatorial constructions of low-density parity check codes for iterative decoding. In *Proceedings IEEE International Symposium on Information Theory*, pages 312–, 2002.

[VDV+10] A. Voicila, D. Declercq, F. Verdier, M. Fossorier, and P. Urard. Low-complexity decoding for non-binary LDPC codes in high order fields. *IEEE Transactions on Communications*, 58(5) :1365–1375, 2010.

[Vit67] A.J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2) :260–269, 1967.

[XCD+07] J. Xu, L. Chen, I. Djurdjevic, Shu Lin, and K. Abdel-Ghaffar. Construction of regular and irregular LDPC codes : Geometry decomposition and masking. *IEEE Transactions on Information Theory*, 53(1) :121–134, 2007.

[ZC11] Xinmiao Zhang and Fang Cai. Reduced-complexity decoder architecture for non-binary LDPC codes. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 19(7) :1229–1238, 2011.

[ZGM+12a] Y. Zrelli, R. Gautier, M. Marazin, E. Rannou, and E. Radoi. Focus on theoretical properties of blind convolutional codes identification methods based on rank criterion. *MTA Review*, XXII(4) :213–234, Dec 2012.

[ZGM+12b] Y. Zrelli, R. Gautier, M. Marazin, E. Rannou, and E. Radoi. Focus on theoretical properties of blind convolutional codes identification methods based on rank criterion. In *2012 9th International Conference on Communications (COMM)*, pages 353–356, 2012.

[ZMGR11] Y. Zrelli, M. Marazin, R. Gautier, and E. Rannou. Blind identification of convolutional encoder parameters over $\mathrm{GF}(2^m)$ in the noiseless case. In *Proceedings of the International Conference on Computer Communication Networks*, Maiu, Hawaii, August 2011.

Liste des publications

Article dans une revue à comité de lecture

- **Y. Zrelli**, R. Gautier, M. Marazin, E. Rannou and E. Radoi. "Focus on theoretical properties of blind convolutional codes identification methods based on rank criterion", *MTA Review*, XXII (4) : 213-234, Dec. 2012, extended version of the paper presented at the 9th IEEE Communications Conference, http ://www.journal.mta.ro/.

Communications avec actes dans un congrés international

- **Y. Zrelli**, R. Gautier, M. Marazin and E. Rannou. "Blind identification of convolutional encoder parameters over GF(2m) in the noiseless case", *IEEE ICCCN 2011*, Maui, Hawaii : États-Unis (2011).

- **Y. Zrelli**, S. Houcke, C. Langlais and M. Ammar. "Blind CFO estimation for OFDM-IDMA system in Rayleigh fading multipath channel", *2nd International Conference on Information Processing and Wireless Systems (IP-WiS)*, Sousse : Tunisia (2012).

- **Y. Zrelli**, R. Gautier, M. Marazin, E. Rannou and E. Radoi. "Focus on Theoretical Properties of Blind Convolutional Codes Identification Methods Based on Rank Criterion", in *special session of Blind Identification Methods for Military Communications and Cognitive Radio Applications for 9th IEEE Communications Conference*, Bucharest : Roumanie (2012).

Communications avec actes dans un congrés national

- **Y. Zrelli**, R. Gautier, M. Marazin, E. Radoi and E. Rannou. "Identification aveugle des paramètres des codes convolutifs non-binaires", Journées Codage et Cryptographie 2011, Saint-Pierre d'Oléron : France (2011).

- **Y. Zrelli**, R. Gautier, M. Marazin et E. Radoi. "Nouvelle méthode robuste d'identification aveugle de la taille des mots de code pour une transmission entachée d'erreurs avec généralisation aux codes correcteurs d'erreurs non-binaires", GRETSI, Brest, 2013.

Articles en cours de soumission ou en préparation

- **Y. Zrelli**, R. Gautier, E. Rannou, M. Marazin and E. Radoi. "Blind Identification of Codewords Size for Non-binary Error Correcting Codes in Noisy Transmission", article en cous de soumission à Eurasip Journal on Wireless Communications and Networking.

- E. Rannou, R. Gautier and **Y. Zrelli**. “Mathematical Formalism and Impact of Mapping/Demapping Functions on Linear Algebra over Galois Field for Digital Communications”, en cous de soumission à IEEE Signal Processing.

- **Y. Zrelli**, R. Gautier, M. Marazin and E. Radoi. “An iterative algorithm for blind identification of error correcting code parity check equations using soft-decision demodulator”, en préparation pour être soumis dans une revue.

Résumé

Mots clés : Contexte non-coopératif - Codes correcteurs d'erreurs - identification aveugle - corps de Galois - décision souple - mapping - demapping

La première partie de ce mémoire porte sur l'identification aveugle des codes correcteurs d'erreurs non-binaires, travaillant dans le corps de Galois GF(2^m). Une étude sur les propriétés des corps de Galois et des codes non-binaires a été conduite afin d'obtenir les éléments indispensables à la mise en oeuvre des méthodes d'identification aveugle. A partir de la seule connaissance des symboles reçus, nous avons développé des méthodes permettant d'identifier les paramètres des codes non-binaires lors d'une transmission non-bruitée et nous avons mis en évidence la pertinence de cette approche lorsque les paramètres de GF(2^m) utilisés à l'émission sont connus à la réception. Nous avons aussi mené une étude théorique approfondie pour justifier l'utilisation du critère du rang par la plupart des méthodes d'identification existantes. Dans le cas d'une transmission bruitée, nous avons développé trois algorithmes dédiés à l'identification en aveugle de la taille des mots de code pour des codes binaires et non-binaires. Pour identifier une base du code dual, nous avons généralisé une technique existante pour les codes binaires, basée sur l'utilisation d'un démodulateur à décision ferme, au cas des codes non-binaires. Puis, nous avons amélioré les performances de détection de cette technique en introduisant un processus itératif basé sur l'utilisation conjointe d'un démodulateur à décision souple et d'un algorithme de décodage à décision souple. Dans la deuxième partie de ce mémoire, nous avons tout d'abord proposé un formalisme mathématique pour étudier l'impact des fonctions de mapping-demapping sur la manipulation des données d'un corps de Galois dans le cas des codes non-binaires. Ensuite, nous avons exploité ce formalisme pour détecter et corriger quelques défauts de transmission. Enfin, nous avons étudié l'impact de certaines fonctions de mapping-demapping sur l'identification aveugle des paramètres des codes non-binaires.

Abstract

Keywords : Non-cooperative context - error correcting codes - blind identification - Galois field - soft decision - mapping - demapping

In the first part of this thesis, we have focused on the blind identification of non-binary error correcting codes over the Galois field GF(2^m). A study of the properties of Galois fields and non-binary codes has been presented so as to get the essential elements for a blind identification of non-binary codes parameters. From the knowledge of only the received symbols, methods have been developed to identify the code parameters in the case of a noiseless transmission. The relevance of this approach has been highlighted when the parameters of the used Galois field are known by the receiver. A theoretical study of rank criterion behaviors has been also presented to justify its use by the most existing identification methods. Then, three blind identification methods of the codeword size for binary and non-binary linear codes have been developped in the framework of a noisy transmission. In order to identify a dual code basis, an existing method for binary codes based on the use of a hard decision demodulation has been generalized to non-binary codes. The detection performance of this technique has been improved when an iterative process based on the joint use of a soft-decision demodulator and a soft-decision iterative decoding is introduced. In the second part of this thesis manuscript, a mathematical formalism is proposed in order to investigate the impact of mapping-demapping functions on linear algebra computations and properties over Galois field in the case of non-binary error correcting codes. Finally, this formalism has been exploited to detect or/and correct some digital transmission problems such as a bad synchronization. Finally, we have studied the impact of some mapping-demapping functions on the blind identification of non-binary error correcting codes parameters.