Les codes algébriques principaux et leur décodage

Daniel Augot

Les codes algébriques principaux et leur décodage

Daniel Augot

Table des matières

Chapitre 1

Introduction

1.1 Correction d'erreur

Le sujet de la théorie des codes correcteurs d'erreur est la transmission fiable d'informations sur un canal de transmission bruité, en utilisant des objets combinatoires et algorithmiques appelés *codes correcteurs d'erreurs*.

Dans un mécanisme de transmission, il y a trois entités : l'émetteur, le récepteur et le canal de transmission. L'objectif de l'émetteur est de communiquer au récepteur un *message* m. Le canal de transmission bruité est capable de communiquer des suites arbitrairement longues de symboles d'un alphabet Σ (un des cas les plus importants étant $\Sigma = \{0, 1\}$). Alors l'espace $\mathcal{M}$ des messages à coder est Σ^k, l'ensemble des suites de symboles de longueur k. On note les mots $m = (m_0, \ldots, m_{k-1})$ « horizontalement ».

Émetteur et récepteur se mettent d'accord sur la longueur n des suites codées à transmettre, appelée la *longueur du code*, les messages échangés appartenant donc à Σ^n, que l'on appellera l'*espace ambiant*. L'émetteur et le récepteur se mettent aussi d'accord sur une *fonction de codage*, E, injective : $E : \Sigma^k \to \Sigma^n$, utilisée pour coder les messages avant transmission. L'image $C = \{E(m), \quad m \in \mathcal{M}\}$ est appelé le *code*. Le rapport k/n, noté R, est le *taux de transmission* ou *rendement* du code, c'est le premier paramètre fondamental d'un code.

Prescrivons maintenant une structure de corps fini sur l'alphabet Σ, $\Sigma = \mathbb{F}_q$, ce qui induit une structure de $\mathbb{F}_q$-espace vectoriel sur Σ^n. On se restreint aux *codes linéaires* c'est-à-dire à l'image par une application linéaire de $\mathbb{F}_q^k$ dans $\mathbb{F}_q^n$, que l'on supposera toujours non singulière. Dans ce cas, un code linéaire est un $\mathbb{F}_q$-sous-espace vectoriel de $\mathbb{F}_q^n$. On spécifiera dorénavant un code linéaire C par sa *matrice génératrice*, qui est une matrice dont les lignes forment une base de C.

Le mot de code $c = E(m)$ étant émis, le canal de transmission produit un vecteur de bruit $e \in \Sigma^n$, et le mot reçu est $y = E(m) + e$. Le récepteur utilise alors une fonction de décodage $D : \Sigma^n \to \mathcal{M}$. Le décodage D doit être rapide, et être tel que $D(y) = m$, avec grande probabilité. Intuitivement, le code introduit une redondance en augmentant la longueur des messages, et cette redondance est utilisée pour décoder le message transmis, même s'il est bruité. Du point de vue de la fiabilité de la transmission, la question fondamentale de la théorie de codes est

> Étant donnée une distribution de probabilité P sur le canal de transmission (i.e. une distribution de probabilité sur les erreurs de transmission), quelles sont les meilleures fonctions de codage et de décodage, c'est-à-dire quelle est la plus petite probabilité d'erreur
>
> $$\min_{E,D}\left\{\mathbf{E}_{m\in\mathcal{M}}\left(\Pr_{e\in P}[D(E(m)+e)\neq m]\right)\right\}$$
>
> où $\mathbf{E}$ désigne l'espérance mathématique.

Shannon [42] a étudié les propriétés asymptotiques de cette quantité quand la distribution du bruit sur Σ^n est la distribution produit d'une distribution sur Σ. Dans ce contexte, il existe une quantité $C_0 \in [0,1]$, dépendant du canal, telle que pour tout $R < C_0$ et $\epsilon > 0$, et, pour n assez grand, il existe toujours un couple codage/décodage avec un code de taux R tel que la probabilité d'erreur soit au plus ϵ.

Dans le cadre de cette présentation, nous ne considérerons uniquement le cas du *canal q-aire symétrique*, défini de la manière suivante : chaque symbole de Σ transmis est préservé avec une certaine probabilité $1-\delta$, ou bien est transformé en autre symbole parmi les $q-1$ autres possible avec probabilité $\delta/(q-1)$, les événements étant indépendants d'un symbole à l'autre.

D'un autre coté, Hamming a défini les notions de *code correcteur d'erreur* et de code *détecteur d'erreur* [26]. Définissons le *poids de Hamming* d'une séquence $x \in \Sigma^n$ comme le nombre de composantes non nulles de x, et la *distance de Hamming* entre x et y comme le poids de la différence $x-y$ (c'est-à-dire le nombre de composantes où x et y diffèrent). C'est bien une distance. On définit alors la distance minimale d'un code C comme la plus petite distance entre deux mots distincts du code C. Le canal de transmission crée en général un vecteur e de petit poids, par exemple de poids borné par t. On dira qu'un code correcteur corrige t erreurs si les boules de rayon t centrées sur les mots de code ne s'intersectent pas. En effet si le poids de l'erreur est inférieur à t, alors, si C est t-correcteur, il y a unicité du mot de code le plus proche. Une capacité de correction t implique que la distance minimale entre deux mots distincts du code est supérieure à $2t+1$.

On parle alors de *décodage unique* car il y a unicité du mot de code à distance t du mot reçu. Si toutefois on considère des rayons $\tau > t$, avec $t\lfloor\frac{d-1}{2}\rfloor$, il n'y a plus unicité des mots de code à distance τ du mot reçu. Dans ce cas on demande au décodeur de retourner la liste des mots à distance τ, et on parle de *décodage en liste*.

La distance minimale est le deuxième paramètre fondamental d'un code. On parlera d'un code $[n,k,d]_q$ pour un code de longueur n, de dimension k et de distance minimale d, défini sur le corps $\mathbb{F}_q$. Du point de vue de Hamming, la question fondamentale est

> Étant donné un alphabet Σ de taille q, et deux entiers n et k, $k < n$, quelle est la plus grande distance minimale relative d/n d'un code $C \subseteq \Sigma^n$ de taux de transmission k/n ?

En effet, une distance minimale élevée induit que le code est capable de corriger des erreurs de poids élevé. L'objectif d'avoir une bonne distance minimale et l'objectif d'avoir une grande dimension sont antagonistes : quand la dimension croît, le nombre de mots de code augmente, et la distance minimale

diminue. Signalons immédiatement que la réponse au problème de Hamming n'est pas connue quand la taille de l'alphabet est petite. Il y a une réponse très satisfaisante à la question quand $q > n$: les codes de Reed-Solomon, qui sont ubiquitaires dans le domaine.

1.2 Plan

Le chapitre suivant présente les codes de Reed-Solomon, ainsi que des variantes (codes de Reed et Muller, codes géométriques), ainsi que leurs propriétés métriques. En ce qui concerne les codes géométriques, je ne va pas refaire toute la géométrie des courbes sur les corps finis, ce dont je ne serai pas capable. Je vais donc utiliser un modèle simplifié de courbes, qui correspond bien par exemple avec les courbes dites hermitiennes qui sont les plus étudiées en codage.

Une fois ces codes introduits, je présenterai dans le troisième chapitre les techniques de décodage par syndrome, qui consistent à retrouver l'erreur à partir de son syndrome, qui peut être calculé à partir du mot reçu. Ces méthodes ont culminé dans [36], avec le décodage des codes géométriques jusqu'à la distance de Feng-Rao [13], en utilisant l'algorithme de Berlekamp-Massey-Sakata [37].

Étonnament, Sudan [45], pour les codes de Reed-Solomon, suivi de Shokrollahi et Wasserman [43] pour les codes géométriques, ont introduit le décodage en liste de ces codes, à base de méthodes d'interpolation. Ces travaux ont été complétés par Guruswami et Sudan [19]. Ces algorithmes sont conceptuellement plus simples, et en relaxant l'hypothèse de l'unicité de mot de code, ils permettent de corriger beaucoup plus d'erreurs. Ces techniques ont relancé la recherche sur les codes de Reed-Solomon, notamment du point de vue de la théorie de la complexité. Je consacrerai la quatrième partie à ces algorithmes.

Chapitre 2

Les codes algébriques principaux

Dans ce chapitre, je présente les codes algébriques les plus simples, obtenus par évalutation : les codes de Reed-Solomon, puis les codes de Reed-Muller. Ces codes ne sont pas satisfaisants vis à vis du problème posé à l'origine par hamming, ce qui motive l'introduction des codes géométriques, du à Goppa [18]. Au cours de chapitre, je donnerai aussi les bornes les plus importantes sur les paramètres de codes (Singleton, Hamming, Varshamov-Gilbert, ainsi que la borne de Tsfasman-Vladut-Zink sur les codes géométriques). En fin de cette partie, je donne les propriétés élémentaires sur les *dvisiseurs sur une courbe*, nécessaires à la description de l'algorithme de Sudan pour les codes géométriques.

2.1 Codes de Reed-Solomon

On considère le corps fini $\mathbb{F}_q$, à q éléments. Soient $x_1, \ldots, x_n \in \mathbb{F}_q$, deux à deux distincts, avec $n < q$,

Définition 1. *La* fonction d'évaluation *associée à* $x_1, \ldots, x_n$ *est*

$$\begin{array}{rccc} \text{ev}: & \mathbb{F}_q[x] & \to & \mathbb{F}_q^n \\ & f & \mapsto & (f(x_1), \ldots, f(x_n)) \end{array}$$

Définition 2 (Codes de Reed-Solomon)**.** *Soient* $x_1, \ldots, x_n \in \mathbb{F}_q$*, avec* $n < q$*, et* $x_i \neq x_j$ *si* $i \neq j$*, et* ev *la fonction d'évaluation. Soit*

$$L_k = \{f \in \mathbb{F}_q[x];\ \deg f < k\},$$

l'ensemble des polynômes de degré inférieur à k *avec* $0 \leq k \leq n$*. Le* code de Reed-Solomon *de dimension* k *est*

$$C = \text{ev}(L_k).$$

Proposition 1. *Le code de Reed-Solomon de longueur* n*, de dimension* k *et de support* $(x_1, \ldots, x_n)$ *est de distance minimale* $d = n - k + 1$*.*

Démonstration. Un polynôme de degré inférieur à k a au plus $k-1$ racines. Son vecteur d'évaluation $\mathrm{ev}(f)$ a donc au moins $n-k+1$ composantes non nulles. D'autre part, considérons par exemple le polynôme $f = f(x) = \prod_{i=1}^{k-1}(X - x_i)$. Alors f est tel que le poids de ev f est exactement $n-k+1$. □

Les codes de Reed-Solomon ont aussi une définition duale, c'est-à-dire, par des relations linéraires satisfaites par les mots de codes. On considère le code de Reed-Solomon défini sur $\mathbb{F}_q$, de longueur $n = q-1$, de support $\{x_1, \ldots, x_n\}$, et de dimension k. On suppose que

$$(x_1, \ldots, x_n) = (\alpha^0, \alpha^1, \alpha^2, \ldots, \alpha^{n-1}), \tag{2.1}$$

dans cet ordre exactement. On identifie le mot $c = (c_0, c_1, \ldots, c_{n-1})$ au polynôme $c(x) = c_0 + c_1 x + \cdots + x^{n-1}$.

Théorème 1. *Soit n impair et α une racine n-ième de l'unité sur $\mathbb{F}_2$. Un mot $c = c(x)$ est dans le code de Reed-Solomon, de longueur n, de dimension k et de support*

$$(\alpha^0, \alpha^1, \alpha^2, \ldots, \alpha^{n-1})$$

si et seulement si $c(\alpha^i) = 0$, pour $i \in \{1, \ldots, n-k\}$.

Démonstration. Soit $c = \mathrm{ev}\, f$, le polynôme correspondant $c(x)$ est

$$c(x) = \sum_{i=0}^{n-1} f(\alpha^i) x^i.$$

On a, par les propriétés de la transformée de Fourier discrète :

$$c(\alpha^j) = n f_{n-j}$$

Puisque f est de degré strictement inférieur à k, les coefficients $f_k \ldots f_{n-1}$ sont nuls, et donc $c(\alpha^j) = 0$ pour $j = 1 \ldots n-k$. □

Une manière synthétique d'énoncer la proposition précédente est d'utiliser la notion de dual d'un code. Pour cela, on définit le produit scalaire standard sur $\mathbb{F}_q^n$.

Définition 3. *Le* produit scalaire standard *de $x, y \in \mathbb{F}_q^n$ est*

$$x \cdot y = \sum_{i=1}^{n} x_i y_i.$$

Le code dual $C^\perp$ *de C est $C^\perp = \{y \in \mathbb{F}_q^n;\ x \cdot y = 0;\ \forall x \in C\}$.*

Proposition 2. *Le dual du code de Reed-Solomon de dimension k de support $(\alpha^0, \alpha, \alpha^2, \ldots, \alpha^{n-1})$ est le code de Reed-Solomon de dimension $n-k$ de même support.*

Démonstration. Soit C_k le code de Reed-Solomon de dimension k de support $(\alpha^0, \alpha, \alpha^2, \ldots, \alpha^{n-1})$. Soit $c = (c_0, \ldots, c_{n-1})$ un mot de C_k, on a $c(1) = \cdots = c(\alpha^{n-k}) = 0$, soit matriciellement :

$$\begin{bmatrix} 1 & \ldots & 1 \\ \alpha_0^1 & \ldots & \alpha_{n-1}^1 \\ \vdots & & \vdots \\ \alpha_1^{n-k-1} & \ldots & \alpha_{n-1}^{n-k-1} \end{bmatrix} c^t = 0$$

Les lignes de cette matrice sont $\mathrm{ev}(1), \dots \mathrm{ev}(x^{n-k-1})$ qui forment une base du code de Reed-Solomon de dimension $n-k$. □

On a donc deux manières de définir les codes de Reed-Solomon : soit comme codes d'évaluation, où les mots de code sont des vecteurs d'évaluation de polynômes de degré contrôlé, soit comme dual de codes d'évaluation. Cette dualité nous permettra de concevoir des algorithmes de décodage de natures très différentes, suivant que le code est vu comme code d'évaluation, ou comme codes définis par des syndromes. C'est la base de ma présentation : le chapitre suivant présente le décodage des codes définis par syndrome, celui qui le suit le décodage des codes définis par évaluation.

2.2 Bornes sur les codes

Proposition 3 (Borne de Singleton)**.** *Tout code linéaire C de paramètres $[n, k, d]$ vérifie $k + d \leq n + 1$.*

Démonstration. Soit $h_1, \dots, h_{n-k}$ une base de $C^\perp$, et H la matrice $(n-k) \times n$ dont les lignes sont $h_1, \dots, h_{n-k}$. Alors, soit $x = (x_1, \dots, x_n)$ un mot du code C, de poids w, on a $Hx^t = 0$. Autrement dit, les positions $i_1, \dots, i_w$ non nulles de x, ainsi que les coefficients $x_{i_1}, \dots, x_{i_w}$ de x imposent une relation de dépendance linéaire sur les colonnes $c_1, \dots c_n$ de H du type :

$$x_{i_1} c_{i_1} + \cdots + x_{i_w} c_{i_w} = 0.$$

On a donc que la distance minimale de C est d si et seulement tout sous ensemble de $d-1$ colonnes de H est indépendant, et s'il existe une famille libre de d colonnes de H. Maintenant, le rang de H est $n-k$ et c'est une borne supérieure le nombre maximal de colonnes indépendantes. Donc

$$d - 1 \leq n - k.$$

□

Définition 4. *On dit qu'un code $[n, k, d]_q$ vérifiant $k + d = n + 1$ est* MDS (Maximum Distance Separable).

Les codes de Reed-Solomon atteignent cette borne et sont donc MDS. Est-ce la fin de la théorie des codes ? Non, car la borne MDS est très simple, et fait pas intervenir la taille de l'alphabet. On voit que les codes de Reed-Solomon présenté ici vérifient $n \leq q - 1$. Ils présentent donc le défaut majeur d'être de longueur bornée, à alphabet fixé.

Il y a des astuces permettant d'augmenter de deux la longueur des codes de Reed-Solomon sur des corps de caractéristique 2 [31, Chapitre 11]. On se sait pas si on peut faire mieux. Les résultats connus sont rassemblés dans la « conjecture MDS » :

Conjecture 1 (Conjecture MDS)**.** *Tous les codes MDS vérifient $n \leq q + 1$, sauf si $k = 3$ ou $k = q - 1$, avec q pair, auquel cas on a $n \leq q + 2$.*

Il faut donc des bornes qui fassent intervenir la taille de l'alphabet. La plus simple de ces bornes est la borne de Hamming. Soit $V_q(n,t)$ le nombre de mots dans une boule de rayon t dans $\mathbb{F}_q^n$. On a :

$$V_q(n,t) = \sum_{i=0}^{t} (q-1)^i \binom{n}{i}.$$

Proposition 4 (Borne de Hamming)**.** *Soit $A_q(n,d)$ le plus grand cardinal d'un code q-aire de longueur n, et de distance minimale d. Alors :*

$$A_q(n,d) \leq \frac{q^n}{V_q(n,t)}$$

où $t = \lfloor \frac{d-1}{2} \rfloor$.

Démonstration. Les boules des rayons t, centrées sur les mots du code sont disjointes. Il y en a au plus $A_q(n,d)$. □

Lorsque le rapport t/n est constant, rappelons qu'on a l'équivalent, quand n croît :

$$\frac{\log_q V_q(n,t)}{n} \approx H_q(t/n),$$

où $H_q(x)$ est la fonction d'entropie q-aire, définie par

$$H_q(x) = x \log_q(q-1) - x \log_q x - (1-x) \log_q(1-x).$$

La borne de Hamming prend alors la forme asymptotique suivante :

$$R \leq 1 - H_q(\delta/2)$$

où $R = k/n$ est le taux de transmission, et $\delta = d/n$ est la distance minimale relative. La borne de Singleton donne $R \leq 1-\delta$. Les graphiques 2.1 montre l'écart entre ces deux bornes. Il existe d'autres bornes supérieures, plus pertinentes que la borne de Hamming et la borne de Singleton (bornes de Bassalygo-Elias, de Plotkin, de McEliece-Rodemich-Rumsey-Welch), voir [31].

Nous avons les paramètres suivants, qui borne inférieurement les paramètres des meilleurs codes. On dit aussi que c'est une borne d'existence.

Proposition 5 (Borne de Gilbert-Varshamov)**.** *Si*

$$q^k V_q(n, d-1) < q^n,$$

alors il existe un code $[n,k,d]_q$.

La forme asymptotique, quand n tend vers l'infini, est la suivante.

Théorème 2 (Borne de Varshamov-Gilbert asymptotique)**.** *Si $R \leq 1 - H_q(\delta)$, il existe une famille de codes linéaires de longueur n, de dimension k et de distance minimale, telle que*

$$\lim_{n\to\infty} k/n = R \text{ et } \lim_{n\to\infty} d/n = \delta.$$

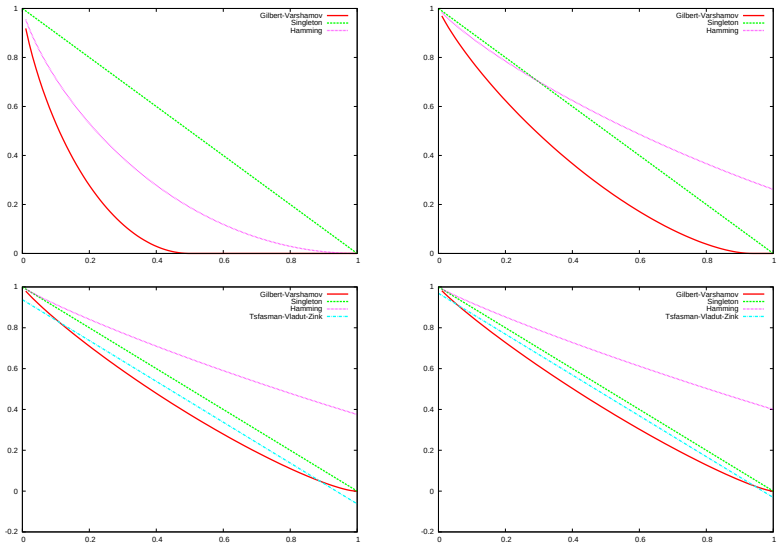

FIGURE 2.1 – Comparaison des bornes asymptotiques de Singleton, de Hamming, et de Varshamov-Gilbert pour les corps à deux éléments et à 16 élements (figures du haut), 256 et 1024 éléments (figures du bas). On voit que pour le corps à deux éléments, ces bornes sont distantes, et que pour le « grand » corps, les bornes de Singleton et de Varshamov-Gilbert s'écrasent, alors que la borne de Hamming ne devient plus significative. Dans les deux courbes du bas, est aussi representée la borne de Tsfasman-Vladut-Zink, qui est meilleure que celle de Varshamov-Gilbert, et on voit que la borne de Tsfasman-Vladut-Zink se rapproche de la borne de Singleton

On en a réalité plus : on peut monter que si on tire un code de longueur n et de dimension k, alors sa distance minimale sera telle que $R \approx 1 - H_q(\delta)$, avec probabilité tendant vers 1, quand n tend vers l'infini. Il est notable que la borne de Varshamov-Gilbert a longtemps été conjecturée comme une borne supérieure. Cependant, il a récemment été montré dans le cas binaire qu'il existe des codes meilleurs que ceux prédits par cette borne [16]. Dans le cas de corps de cardinal plus élevé, il existe de codes de paramètres au dessus de la borne de Varshamov-Gilbert. Ces codes dit géométriques sont obtenus par la construction de Goppa [18]. Cela a motivé tous les travaux sur les codes géométriques, et notamment sur l'obtention d'algorithmes de décodage.

2.3 Codes géométriques

2.3.1 Codes « multivariés »

Comme on l'a vu, les codes de Reed-Solomon sont MDS : leur dimension k et leur distance minimale vérifient $k + d = n + 1$, n étant leur longueur, ces paramètres étant les meilleurs possibles. Leur principal défaut est d'être défini sur un alphabet de taille q supérieure à n : à alphabet fixé, on ne peut pas avoir des codes MDS de longueur croissante. Une manière naturelle est de passer dans un contexte multivarié.

On va donc considérer le corps $\mathbb{F}_q$, et une énumération $\{P_1, \ldots, P_{q^m}\}$ des points de $\mathbb{F}_q^m = \{P_1, \ldots, P_{q^m}\}$. Dans un premier temps, nous notons $n = q^m$, et nous considérons tous les points de $\mathbb{F}_q^m$.

Définition 5. *La fonction d'évaluation associée aux points $\{P_1, \ldots, P_n\}$ est*

$$\begin{array}{rccc} \mathrm{ev}: & \mathbb{F}_q[X_1, \ldots, X_m] & \to & \mathbb{F}_q^n \\ & f & \mapsto & (f(P_1), \ldots, f(P_n)) \end{array}$$

Définition 6 (Codes de Reed-Muller généralisés)**.** *Soit $\{P_1, \ldots, P_n\}$ les points de $\mathbb{F}_q^m$ et* ev *la fonction d'évaluation associée. On considère l'espace*

$$L_r = \{f \in \mathbb{F}_q[X_1, \ldots, X_m];\ \deg f \leqslant r\}.$$

Le code de Reed-Muller *d'ordre r est*

$$C = \mathrm{ev}(L_r),$$

où $0 \leqslant r \leqslant m(q-1)$.

En comptant le nombre de monômes de degré inférieur à r, et en utilisant le lemme de Schwartz-Zippel [52, 41] sur le nombre de zéros d'un polynôme multivarié sur un corps fini, on peut déterminer la dimension et la distance minimale des codes de Reed-Muller.

Proposition 6 (Lemme de Schwartz-Zippel)**.** *Soit $f \in \mathbb{F}_q[X_1, \ldots, X_m]$ de degré total au plus égal à r. Alors le nombre de zéros de f sur $\mathbb{F}_q^m$ est borné par rq^{m-1}.*

Corollaire 1. *Pour $r \leqslant q-1$, le code de Reed-Muller d'ordre r a pour dimension $\binom{r+m}{m}$ et $(q-r)q^{m-1}$ comme distance minimale.*

Pour r plus élevé, la taille du corps intervient en ligne de compte. Soit $r = u(q-1) + s < m(q-1)$ la distance minimale du code de Reed-Muller d'ordre r est $(q-s)q^{m-r-1}$, et sa dimension est

$$\sum_{i=0}^{r}\sum_{k=0}^{m}(-1)^k\binom{m}{k}\binom{i-kq+m-1}{i-kq}$$

voir [2]. Dans tous les cas, on peut borner comme suit les paramètres du code de Reed-Muller généralisé : $k \leqslant (r+1)^m$, et $d = q^m(1-\frac{r}{q})$. En rapportant tout à la longueur $n = q^m$, on obtient pour taux de transmission

$$R = \frac{k}{n} \leqslant \left(\frac{r+1}{q}\right)^m$$

et, pour distance minimale relative :

$$\delta = \frac{d}{n} = 1 - \frac{r}{q}.$$

On voit que les performances sont très mauvaises par rapport à la borne de Singleton, qui donnerait $R \approx \frac{r}{q}$: quand le nombre de variables augmente, le taux de transmission se dégrade exponentiellement avec le nombre de variables. Notons toutefois que le code de Reed-Muller binaire d'ordre un en cinq variables (petite dimension, grande distance minimale) a été utilisé en 1972 par la sonde spatiale Mariner pour envoyer des photos de Mars [31].

2.3.2 Codes définis par courbes algébriques

L'idée naturelle est de considérer des polynômes multivariés que l'on va évaluer sur des points $P_1, \ldots, P_n$ de $\mathbb{F}_q^m$, avec $n < q^m$, et où les points $P_1, \ldots, P_n$ sont bien choisis. Cela revient à considérer des codes plus courts, de longueur $n < q^m$: les paramètres n, k et d vont décroitre, mais on espère que le taux $R = k/n$, et la distance minimale relative $\delta = d/n$ vont devenir bons. On va donc considérer la fonction d'évaluation :

$$\begin{array}{rccc} \mathrm{ev}: & \mathbb{F}_q[x_1,\ldots,x_m] & \to & \mathbb{F}_q^n \\ & f & \mapsto & (f(P_1),\ldots,f(P_n)) \end{array},$$

et le code associé est

$$C = \mathrm{ev}(L_r),$$

où $L_r = \{f \in \mathbb{F}_q[x];\ \deg f < r\}$. Toute la difficulté est de déterminer la distance minimale et la dimension de ces codes, en fonction des points $P_1, \ldots, P_n$ choisis dans le support, et de bien choisir ces points.

Afin d'avoir de pouvoir se reposer sur une théorie mathématique, qui permette d'avoir des théorèmes pour obtenir des bornes, on va prendre les points $P_1, \ldots, P_n$ sur une courbe $\mathcal{C}$ définie sur $\mathbb{F}_q$. On va aussi devoir prendre des espaces différents pour L_r, car la notion de degré va être aussi modifiée.

Afin de ne pas refaire toute la théorie des variétés algébriques sur les corps finis, et en particulier des courbes, on ne considère que des courbes très particulières, dites en « *position spéciale* » [35]. Cela simplifie grandement la présentation, et permet d'aller tout de suite aux aspects liés au calcul formel des codes

géométriques. De plus, pour les codes considérés en pratique (codes hermitiens), les courbes se présentent naturellement sous cette forme là. Nous supposons connu le livre de Cox, Little et O'Shea [12].

Soit $I \subseteq \mathbb{F}_q[x_1, \ldots, x_m]$ un idéal radical et V la variété associé, qui est l'ensemble des zéros communs des polynômes de I. On suppose V de dimension 1, irréductible, lisse dans sa partie affine, ce qui permet de définir le corps $\mathbb{F}_q(V) = \operatorname{Frac} \mathbb{F}_q[V]$ des fonctions rationnelles, qui est le corps des fractions de $\mathbb{F}_q[x_1, \ldots, x_m]/I$, noté lui même $\mathbb{F}_q[V]$. La variété V est une courbe affine si le degré de transcendance de $\mathbb{F}_q(V)$ est égal à un. L'hypothèse cruciale est que la courbe n'a qu'un seul point à l'infini noté Q.

Exemple 1 (Courbe hermitienne)**.** *Dans le cas de la courbe définie sur $\mathbb{F}_q$ par $x^{q_0+1} = y^{q_0} - y$, avec $q = q_0^2$, il y a un point à l'infini $Q = (0 : 1 : 0)$, et q_0^3 points affines dans $\mathbb{F}_q$. Cette courbe est dite hermitienne, car, après changement de variable, elle admet une équation $x^{q_0+1} + y^{q_0+1} + z^{q_0+1} = 0$, ce qui correspond à un produit scalaire « hermitien » sur $\mathbb{F}_{q_0^2}^3$, pour l'isomorphisme $x \mapsto x^{q_0}$.*

Nous rappelons le théorème suivant.

Théorème 3. *Soit $\mathcal{C}$ une courbe lisse irréductible projective, et f une fonction rationnelle sur $\mathcal{C}$. Alors, comptés avec multiplicités, f autant de zéros que de pôles (dans la clôture algébrique de $\mathbb{F}_q$).*

Ainsi, dans ce contexte simplifié, un polynôme p définit une fonction f sur la courbe affine $\mathcal{C}$. Comme ce polynôme n'a pas de pôle sur la courbe affine, il admet nécessairement comme pôle le point à l'infini Q. L'ordre au pôle de f est égal au nombres de zéros affines de p sur la courbe.

On peut montrer la proposition suivante.

Proposition 7. *Soit $\mathcal{C}$ une courbe affine dans $\mathbb{A}^m(\mathbb{F}_q)$ en position spéciale comme précédemment. Il existe $o_1, \ldots, o_m \in \mathbb{N} \setminus \{0\}$ tel que, pour tout monôme $m = x_1^{i_1} \cdots x_m^{i_m}$, l'ordre au pôle de m en Q est égal à*

$$v_Q(x_1^{i_1} \ldots x_m^{i_m}) = -\left(o_1 i_1 + \cdots + o_m i_m\right).$$

On définit l'ordre monomial $<$ défini par le degré pondéré

$$\operatorname{wdeg}_{o_1,\ldots,o_m}(i_1, \ldots, i_m) = o_1 i_1 + \cdots + o_m i_m,$$

raffiné par l'ordre lexicographique $x_1 < \cdots < x_m$. La proposition suivante nous permet de représenter les fonctions définies sur la courbe affine, ainsi que de déterminer leur ordre au pôle.

Proposition 8. *Soit $\mathcal{C}$ une courbe en position spéciale, et $<$ l'ordre monomial associé. Soit $f \in \mathbb{F}_q[x_1, \ldots, x_m]$, et $\bar{f}$ la fonction associée dans $\mathbb{F}_q[\mathcal{C}]$. Soit G une base de Gröbner de $I(\mathcal{C})$ pour l'ordre $<$, et soit $N(f)$ la forme normale de f relativement à G. Alors l'ordre au pôle Q de $\bar{f}$ est égal à*

$$v_Q(x_1^{i_1} \cdots x_m^{i_m}) = -\left(o_1 i_1 + \cdots + o_m i_m\right)$$

où le terme de tête de $N(f)$ s'écrit $c x_1^{i_1} \cdots x_m^{i_m}$.

Définition 7. *Soit $\mathcal{C}$ une courbe en position spéciale, et $<$ l'ordre monomial associé. Soit $m = x_1^{i_1} \cdots x_m^{i_m}$, le* poids *de m est $\operatorname{wdeg}_{o_1,\ldots,o_m}(i_1, \ldots, i_m) = o_1 i_1 + \cdots + o_m i_m$. Le poids d'un polynôme f est le poids du terme de tête de sa forme normale relativement à une base de Gröbner définissant $I(\mathcal{C})$.*

Exemple 2 (Courbe hermitienne)**.** *Soit $\mathcal{C}$ la courbe $x^{q_0+1} = y^{q_0} - y$, avec $q = q_0^2$, le point à l'infini est $(1:0:0)$. On a $-v_Q(x) = q_0$ et $-v_Q(y) = q_0 + 1$. Soit $I = I(\mathcal{C})$. L'ensemble*

$$\left\{x^i y^j + I | \quad 0 \leqslant i, \quad 0 \leqslant j \leqslant q_0\right\}$$

est une base de l'espace vectoriel $\mathbb{F}_q[\mathcal{C}]$. On définit le degré pondéré de $x^i y^j$ par

$$\mathrm{wdeg}(x^i y^j) = q_0 i + (q_0 + 1)j.$$

Nous pouvons maintenant définir les codes géométriques (à un point) définis sur de telles courbes. Tout d'abord nous devons définir les espaces $L = L(rQ)$ des fonctions à évaluer.

Définition 8. *Soit $\mathcal{C}$ une courbe affine en position spéciale, et Q la place à l'infini. L'*espace associé *à rQ, noté $L(rQ)$ est l'ensemble des fonctions $\bar{f} \in \mathbb{F}_q(\mathcal{C})$ n'admettant qu'un unique pôle en Q, d'ordre au plus r.*

C'est exactement l'espace des formes normales $\bar{f}$ des polynômes f relativement à une base de Gröbner de $I(\mathcal{C})$, telles que $\mathrm{wdeg}(\mathrm{lead}(\bar{f})) \leq r$.

Définition 9. *Soit $\mathcal{C}$ une courbe affine en position spéciale, $P_1, \ldots, P_n$, n points rationels affines distincts. On définit la fonction d'évaluation*

$$\begin{array}{rccc} \mathrm{ev}: & \mathbb{F}_q[\mathcal{C}] & \to & \mathbb{F}_q^n \\ & f & \mapsto & (f(P_1), \ldots, f(P_n)) \end{array}.$$

On note $D = P_1 + \cdots + P_n$ la somme formelle des points $P_1, \ldots, P_n$.

Définition 10. *Soit $\mathcal{C}$ une courbe comme précédemment, et Q la place à l'infini, et $D = P_1 + \cdots + P_n$, où $P_1, \ldots, P_n$ sont n points rationnels affines de $\mathcal{C}$. Le* code géométrique *$C_L(D, rQ)$ est*

$$\mathrm{ev}(L(rQ)).$$

Le code géométrique *$C_\Omega(D, rQ)$ est le dual de $C_L(D, rQ)$.*

Soit $C_L(D, rQ)$ un code défini sur une courbe $\mathcal{C}$, définissant un ordre monomial $\leqslant$ défini par le degré pondéré $\mathrm{wdeg}_{o_1,\ldots,o_m}(x_1^{i_1} \ldots x_m^{i_m}) = o_1 i_1 + \cdots + o_m i_m$ raffiné par l'ordre lexicographique. Une base de l'espace $L(rQ)$ est donnée par les formes normales de monômes $x_1^{i_1} \ldots x_m^{i_m}$ tels que $\mathrm{wdeg}(x_1^{i_1} \ldots x_m^{i_m}) \leq r$. Cet espace n'est pas de dimension r : il existe des degrés w tels qu'il n'existe pas de monômes $x_1^{i_1} \cdots x_m^{i_m}$ vérifiant $\mathrm{wdeg}(x_1^{i_1} \cdots x_m^{i_m}) = w$.

Définition 11. *Soit $\mathcal{C}$ une courbe comme précédemment, Q la place à l'infini, et $\mathrm{wdeg}_{o_1,\ldots,o_m}$ le degré pondéré induit par Q. Le* semi-groupe de Weierstrass *G_Q en $\mathcal{C}$ est l'ensemble des poids des monômes de $\mathbb{F}_q[X_1, \ldots, X_m]$.*

On peut montrer que pour un entier $u \in \mathbb{N}$ assez grand $u + \mathbb{N} \subset G_Q$.

Exemple 3. *Soit la courbe hermitienne $\mathcal{C}$ définie sur $\mathbb{F}_{16}$ par $x^5 = y^4 + y$, on a $-v_\mathcal{C}(x) = 4$ et $-v_\mathcal{C}(y) = 5$. On voit que le semi-groupe de Weierstrass G_Q, qui est égal à $4\mathbb{N} + 5\mathbb{N}$ est*

$$\{0, 4, 5, 8, 9, 10, 12, 13, 14, 15, 16, 17, \ldots\}$$

et il manque $1, 2, 3, 6, 7, 11$.

Définition 12. *Soit $\mathcal{C}$ une courbe comme précédemment, et Q la place à l'infini. Le* genre *g de $\mathcal{C}$ est le cardinal de $\mathbb{N} \setminus G_Q$.*

Théorème 4. *Pour $r > 2g-2$, on a $\dim L(rQ) = r - g + 1$.*

Ce théorème est une conséquence du théorème de Riemann-Roch, voir [44] pour un énoncé et une démonstration valide sur les corps finis.

Proposition 9. *Pour $r > 2g-2$, le code géométrique $C_L(D, rQ)$ est un code $[n, k = r - g + 1, d \geqslant n - r]_q$. Le code géométrique $C_\Omega(D, rQ)$ est un code $[n, n - r + g - 1, d \geqslant r - 2g + 2]_q$.*

On voit que les paramètres k et d des codes géométriques $C_L(P, rQ)$ et $C_\Omega(P, rQ)$ vérifient $k + d \geqslant n + 1 - g$. Ainsi, le défaut de ces codes par rapport à la borne de Singleton est inférieur ou égal au genre de la courbe $\mathcal{C}$.

Exemple 4. *Sur l'alphabet $\mathbb{F}_{256} = \mathbb{F}_{q_0^2}$, on a la courbe hermitienne $x^{q_0+1} - y^{q_0} - y = x^{17} - y^{16} - y$ qui a $q_0^3 = 2^{12} = 4096$ points affines à comparer avec 256 points pour le code de Reed-Solomon défini sur le même alphabet. Le genre de la courbe est $\frac{q_0(q_0-1)}{2} = 120$. On obtient une famille de codes $[n = 4096, k = r+1-120, d \geqslant 4096-r]_{256}$ pour r assez grand. La performance d'un tel code est difficile à évaluer, et il faut pour cela faire des simulations. Dans [49, 1], il est indiqué par simulation qu'on réalise un « gain de codage » en utilisant le code hermitien plutôt qu'un code de Reed-Solomon de même taux de transmission $R = 0,6$.*

À partir, le but sera de construire des courbes avec le plus grand nombre de points sur $\mathbb{F}_q$, et de petit genre. En particulier, on cherche des courbes maximales, relativement au théorème de Hasse-Weil.

Théorème 5 (Borne de Hasse-Weil)**.** *Soit $\mathcal{C}$ une courbe défini sur $\mathbb{F}_q$, de genre g. Le nombre de points $N_q(\mathcal{C})$ sur $\mathbb{F}_q$ vérifie*

$$|N_q(\mathcal{C}) - (q+1)| \leq 2g\sqrt{q} \tag{2.2}$$

Les courbes hermitiennes sont des *courbes maximales* : pour leur genre, ce sont des courbes qui atteigne la borne supérieure de l'intervalle de Hasse-Weil [44].

Du point de vue asymptotique, pour les codes géométriques $C_L(D, rQ)$, on a les rapports suivants :

$$R = \frac{k}{n} = \frac{(r - g + 1)}{n}$$

et

$$\delta = \frac{d}{n} = \frac{n - r}{n} = 1 - \frac{r}{n}.$$

Donc

$$R \approx 1 - \delta - \frac{g}{n}.$$

En notant $N_q(g)$ le nombre maximal de points sur $\mathbb{F}_q$ d'une courbe de genre g, la quantité asymptotique d'intérêt est

$$A(q) = \limsup_{g \to \infty} \frac{N_q(g)}{g}$$

Le résultat difficile mais important de Tsfasman-Vladut-Zink [46] est que si q est un carré, alors $A(q) = \sqrt{q} - 1$. On obtient donc

Théorème 6. *Soit q un carré, et R et δ tels que*

$$R + \delta = 1 - \frac{1}{\sqrt{q} - 1}.$$

Alors il existe une famille de codes $[n, k, d]_q$ telle que $\frac{k}{n} \to R$ et $\frac{d}{n} \to \delta$.

Cette borne est meilleure que la borne de Varshamov-Gilbert quand $q \geq 49$. En particulier, on peut construire de manière explicite des codes meilleurs que les codes aléatoires (pour un alphabet assez grand), avec des beaux objets mathématiques. La construction se fait en temps polynomiale en longueur des codes. Il est aussi à noter que les codes géométriques permettent aussi de construire de bons codes non asymptotiquement longs. À titre d'exemple, on peut obtenir des codes sur $\mathbb{F}_8$, de longueur 64 et 65 battant les meilleurs codes connus, par exemple [10].

2.4 Diviseurs

Pour la troisième partie de ce cours, j'aurai besoin de la notion de diviseur, dont je rappelle la théorie prise dans [44], où la théorie de corps de fonctions à une variable sur les corps finis est proprement faite. Soit $P \in \mathcal{C}$ un point non singulier, l'ensemble $\mathcal{M}$ des fonctions $f \in \mathbb{F}_q[\mathcal{C}]$ telles que $f(P) = 0$ est un idéal principal. Soit t qui engendre $\mathcal{M}$, et soit f telle que $f(P)$ est défini, alors l'ordre en f en P, noté $v_P(f)$ est l'entier m tel que $f = ut^m$, avec $u(P) \neq 0$. Si $f(P)$ n'est pas défini (on dit que P est un pôle de f), alors $f^{-1}(P)$ est défini, et on définit $v_P(f) = -v_P(f^{-1})$.

Un diviseur D est une somme formelle de poinds $\sum_P n_P P$, où les $n_P \in \mathbb{Z}$ sont presque tous nuls. Le degré $\deg D$ d'un diviseur $D = \sum_P n_P P$ est $\sum n_P$.

On définit alors le diviseur $(f)_0$ des zéros de f qui la somme formelle de points :

$$(f)_0 = \sum_{P | f(P)=0} v_p(f),$$

et le diviseur des pôles

$$(f)_\infty = \sum_{P | f^{-1}(P)=0} -v_P(f).$$

Le diviseur (f) associé à f est $(f) = (f)_0 - (f)_\infty$, et on a $\deg(f) = 0$.

Proposition 10. *Dans les diviseurs $(f)_0$ et $(f)_\infty$ n'apparaissent qu'un nombre fini de points. De plus $(f) = 0$ (une fonction a autant de zéros que de pôles, comptés avec multiplicité).*

On définit la relation d'ordre partiel suivante sur les diviseurs $D_1 \geqslant D_2$, si $D_1 = \sum n_P^{(1)} P$, et $D_2 = \sum n_P^{(2)} P$, avec $n_P^{(1)} \geqslant n_P^{(2)}$ pour tout point P.

Définition 13. *Soit D un diviseur, l'*espace associé *à D est l'ensemble de fonctions :*

$$L(D) = \{f;\ (f) + D \geqslant 0\} \cup \{0\}.$$

Alors, le théorème de Riemann-Roch, dont le théorème 4 est une version très simplifiée est le suivant.

Théorème 7. *La dimension de l'espace $L(D)$ associé au diviseur D est finie, et vérifie*

$$\dim L(D) \geq \deg D + 1 - g,$$

où g est le genre de la courbe. On a égalité quand $\deg D > 2g - 2$.

On note aussi que si un diviseur D est de degré négatif alors $L(D) = \{0\}$. Ceci est du au fait que le degré d'un diviseur de fonction est nul : si $\deg D$ est négatif, il ne peut pas y avoir de fonctions (f) telles que $(f) + D \geq 0$.

Chapitre 3

Décodage par syndrome

Dans ce chapitre, je présente ce qui est appelé ≪ le décodage par syndrome ≫. Le principe est d'associer à tout mot reçu après transmission un *syndrome* S. Celui est en réalité déterminé uniquement par l'erreur e de transmission, et si celle ci est de poids faible, alors e est uniquement déterminée par S.

La plupart des codes algébriques se décodent ≪ par syndrome ≫. Ces méthodes amènent à chercher un ou plusieurs polynomes de petit degré dans un idéal que l'on définit avec les syndromes. On retrouve donc des notions familières telles que l'algorithme d'Euclide (dans le cas univarié), ou les bases de Gröbner (dans le cas multivarié).

Toutefois, j'ai choisi de présenter comment l'algorithme de Berlekamp-Massey est en général utilisé pour trouver ces polynomes de petit degré (en fait des polynômes définissant une récurrence linéaire). L'algorithme de Berlekamp-Massey, utilisés dans de nombreux contextes, permet de décoder les codes de Reedq-Solomon et apparentés. Cet algorithme se généralise en plusieurs variables pour donner l'algorithme de Berlekamp-Massey-Sakata, qui permet de décoder les codes géométriques à un point.

3.1 Le principe du décodage par syndrome

Soit C un code linéaire $[n, k, d]_q$, et H une matrice génératrice de son dual. On suppose que la distance minimale de C est $d = 2t+1$, et t est donc la capacité de correction. La matrice H a $n - k$ lignes et n colonnes, et elle caractérise les mots de C. On a donc

$$c \in C \iff Hc^t = 0.$$

Soit c le mot de code émis, *inconnu* du décodeur. On suppose que le mot reçu est $y = c + e$, où e est l'erreur *inconnue* du décodeur, qui vérifie $\mathrm{w}(e) \leqslant t$. Maintenant on a $Hy^t = Hc^t + He^t = He^t$, puisque le mot transmis c est dans C. On appelle S le vecteur $Hy^t = He^t$ le *syndrome* (ou *les syndromes* suivant que l'on parle du vecteur ou ses composants) de l'erreur e.

Soit $\mathcal{B}(0, t)$, la boule centrée en zéro de rayon t. Le problème du décodage est d'inverser la fonction ≪ syndrome ≫.

$$\begin{array}{lcl} \mathcal{B}(0,t) & \to & \mathbb{F}_q^{n-k} \\ e & \mapsto & He^t. \end{array}$$

C'est hélas un problème NP-complet pour les codes linéaires généraux [4] (quand t est remplacé par w quelconque). Ce problème reste tout aussi difficile avec "prétraitement" : si on autorise un temps de calcul infini d'étude de la matrice H, le problème reste difficile [7]. Notons aussi que le problème du calcul de la distance minimale d d'un code linéaire est lui aussi difficile [47].

On se sait résoudre ce problème que dans des cas très particuliers, notamment dans le cas des codes de Reed-Solomon, et des codes géométriques à un point en position spéciale comme ceux présentés précédemment. Notons de même que décoder les codes de Reed-Solomon et les codes géométriques est aussi un problème difficile quand le nombre d'erreurs est évelé [20, 9, pour les codes de Reed-Solomon], et [8, pour les codes géométriques]. Nous verrons dans le chapitre suivant comment encadrer les valeurs pour lesquelles le problème est facile/difficile.

3.2 Décodage des codes de Reed-Solomon

Soit $\mathbb{F}_q$ le corps fini de cardinal q, $n = q - 1$, et α un racine n-ième de l'unité. Nous considérons un code de Reed-Solomon, de longueur n, de support $(\alpha^0, \ldots, \alpha^{n-1})$, de distance minimale $2t + 1$, corrigeant donc t erreurs, avec $2t = n - k$. Un mot de code $c = c_0 + c_1 x + \cdots + c_{n-1}x^{n-1}$ est transmis (et inconnu), le mot $y = y_0 + y_1 x + \cdots + y_{n-1}x^{n-1}$ est reçu, et on considère l'erreur de transmission $e = y - c = e_0 + \cdots + e_{n-1}x^{n-1}$ (inconnue). Dans le cas des codes de Reed-Solomon, les syndromes sont simplement quelques coefficients de la transformée de Fourier.

Définition 14. *Soit C le code de Reed-Solomon de support $(\alpha^0, \alpha^1, \ldots, \alpha^{n-1})$, de dimension k. Sa distance minimale est $d = n - k + 1$ (que nous prenons impaire pour simplifier), et sa capacité de correction est $t = \frac{n-k}{2}$. Soit $y = c + e$ le mot reçu, avec c le mot de code et e l'erreur. Les* syndromes $S_1, \ldots, S_{n-k}$ *de e relativement à C sont*

$$S_1 = y(\alpha), S_2 = y(\alpha^2), \ldots, S_{n-k} = y(\alpha^{n-k}).$$

On a $y \in C$ si et seulement si $S_1 = S_2 = \cdots = S_{n-k} = 0$.

Définition 15. *Soit $e = (e_0, \ldots, e_{n-1})$ un erreur de poids w inférieur ou égal à t. On définit le* support de l'erreur

$$I = \{i \in \{0, \ldots, n-1\} | e_i \neq 0\}$$

et on définit

$$\{X_1, \ldots X_w\} = \{\alpha^i | i \in I\}$$

qui sont les localisateurs d'erreur *de e.*

On décrit l'ensemble des localisateurs d'erreur par le polynôme localisateur d'erreur, comme suit :

Définition 16. *Soit e une erreur de localisateurs $X_1, \ldots, X_w$, le* polynôme localisateur *de e est*

$$\sigma(x) = \prod_{i=1}^{w} (1 - X_i x).$$

Le polynôme évaluateur *de e est*

$$\omega(x) = \sum_{i \in I} e_i \alpha^i \prod_{j \in \{1,\ldots,w\} \setminus \{i\}} (1 - \alpha^j x).$$

Si w est le poids de l'erreur, alors $\deg \sigma(x) = w$, et $\deg \omega(x) \leqslant w - 1$. Si on connait ces deux polynômes, alors on retrouve les X_i par factorisation ou recherche de racines (bien que l'implantation utilise la « recherche de Chien » [11] qui est manière intelligente de faire une recherche exhaustive), et ensuite on trouve e_i, $i \in I$, par la formule :

$$\omega\left(\alpha^{-i}\right) = e_i \alpha^i \sigma'(\alpha^{-i}),$$

où $\sigma'(x)$ est la dérivée de $\sigma(x)$ (formules de Forney [15]).

Donc nous réduisons le problème du décodage des codes de Reed-Solomon à celui de la détermination des polynômes $\sigma(x)$ et $\omega(x)$. Ceux ci sont liés de manière très forte aux syndromes de l'erreur. En effet, on calcule :

$$\begin{aligned}
\frac{\omega(x)}{\sigma(x)} &= \sum_{i \in I} \frac{e_i \alpha^i}{1 - \alpha^i x} \\
&= \sum_{i \in I} e_i \alpha^i \sum_{j=0}^{\infty} \left(\alpha^i x\right)^j \\
&= \sum_{j=0}^{\infty} x^j \sum_{i \in I} e_i \left(\alpha^{j+1}\right)^i \\
&= \sum_{j=0}^{\infty} x^j e(\alpha^{j+1}) \\
&= \sum_{j=1}^{\infty} x^{j-1} e(\alpha^j) \\
&= \sum_{j=1}^{\infty} x^{j-1} S_j.
\end{aligned}$$

Le décodeur est partiellement aveugle : puisque le code est défini par $\alpha, \ldots, \alpha^{2t}$, il connait exactement les $e(\alpha^j) = y(\alpha^j)$ pour $j \in \{1, \ldots, n-k\}$, mais pas les syndromes $e(\alpha^j)$ pour $j > n - k$. Au niveau du décodeur, on peut donc écrire la relation algébrique

$$\frac{\omega(x)}{\sigma(x)} = S(x) \bmod x^{n-k+1} \tag{3.1}$$

avec $S(x) = \sum_{j=1}^{n-k} e(\alpha^j) x^{j-1}$ est la série tronquée des syndromes. Rappelons que $n - k = 2t$, nous avons donc :

Définition 17. *L'*équation clé *du décodage des codes de Reed-Solomon d'un mot y de série tronquée de syndrômes* $S(x) = \sum_{i=1}^{2t} S_i x^i$ *est l'équation*

$$\frac{\omega(x)}{\sigma(x)} = S(x) \bmod x^{2t+1}, \tag{3.2}$$

dont les inconnues sont les polynômes $\sigma(x)$ *et* $\omega(x)$ *de degré inférieurs à* t *et* $t - 1$ *respectivement, avec* $\sigma(x)$ *unitaire.*

3.2.1 Suites récurrentes linéaires

L'équation 3.1 peut être résolue en la considérant comme un système linéaire donc les inconnues sont les coefficients de $\sigma(x)$ et $\omega(x)$. De manière plus efficiace, elle peut aussi être résolue avec l'algorithme d'Euclide. Cette méthode est présenté de manière très synthétique dans [48], et il s'agit de calculer un $(t+1, t)$ approximant de Padé de la série $S(x)$. Toutefois, pour des raisons d'efficacité et d'implantation, les codeurs préfèrent utiliser l'algorithme de Berlekamp-Massey [32, 3]. Berlekamp a introduit cet algorithme à l'origine pour résoudre le problème du décodage des codes de Reed-Solomon, et Massey a montré qu'il servait à résoudre un problème plus large, qui est celui de la synthèse d'une suite linéaire récurrente.

Remarquons que l'équation (3.2) entraîne

$$\sum_{i=0}^{t} \sigma_{j-i} S_i = 0, \quad j = t \dots 2t. \tag{3.3}$$

Comme $\sigma_0 = 1$ par construction, on a les formules de récurrence :

$$S_j = -\sum_{i=1}^{t} \sigma_i S_{j-i}, \quad j = t \dots 2t. \tag{3.4}$$

On voit donc que le polynôme $\sigma(x)$ définit une relation de récurrence satisfaite par la suite $S_1, \dots, S_{2t}$. L'algorithme de Berlekamp-Massey est un algorithme qui, étant donné une suite $S_1, \dots, S_{2t}$, produit la plus petite relation de récurrence linéaire satisfaite par cette suite.

3.2.2 Algorithme de Berlekamp-Massey

L'algorithme de Berlekamp-Massey peut être défini indépendamment du problème de décodage, avec le spécification suivante :

Entrée : $S_1, \dots, S_{2t}$

Sortie : Un polynôme $\sigma = 1 + \sum_{i=1}^{w} \sigma_i x^i$ de degré $w \leq t$ minimal, vérifiant (3.4).

L'algorithme de Berlekamp-Massey est de nature incrémentale : il produit, pour chaque étape j de son exécution, un polynôme σ de degré minimal tel que

$$S_k = -\sum_{i=1}^{\deg \sigma} \sigma_i S_{k-i}, \quad k = \deg \sigma \dots j.$$

Pour passer de l'étape j à l'étape $j+1$, il faut prendre en compte S_{j+1}. À l'étape $j+1$, le polynôme courant σ est testé, en calculant

$$\overline{S}_{j+1} = -\sum_{i=1}^{\deg \sigma} \sigma_i S_{j-i},$$

qui est la *prédiction* du polynôme σ pour S_{j+1}. Si $\overline{S}_{j+1} = S_{j+1}$, la prédiction est correcte, et σ est maintenu tel quel. Sinon, on a une *discrépance*

$$\delta = S_{j+1} - \overline{S}_{j+1},$$

associée à σ. Dans ce cas, on met à jour σ de la manière suivante :

$$\sigma \leftarrow \sigma - \delta\delta_0^{-1}x^l\sigma_0,$$

où σ_0 est un polynôme (maintenu en table) qui a produit la discrépance la plus récente $\delta_0 \neq 0$ dans une étape précécente, et l est la différence de degré de σ et σ_0. On vérifie bien que $\sigma(x)$ donne une relation de récurrence pour $S_1, \dots S_{j+1}$.

Toute la difficulté est de garantir de calculer le polynôme σ de plus petit degré. L'algorithme maintient donc deux polynômes σ et σ_0, qu'il met à jour au fur et à mesure de sa lecture de la séquence S. L'optimisation du polynôme σ_0 auxiliaire est délicate, et la preuve de l'algorithme associé prend généralement deux à trois pages assez pénibles, bien qu'élémentaires. Le pseudo-code de l'algorithme est présenté dans l'algorithme 1.

La complexité de l'algorithme est de $O(t^2)$. Notons que Blahut en présente des versions matricielles, mais aussi des versions ≪ rapides ≫, à condition de disposer de convolutions rapides [6, 5].

Nous obtiendrons l'algorithme de Berlekamp-Massey ainsi que sa preuve comme sous produit de l'algorithme Berlekamp-Massey-Sakata, qui est une version multivariée de l'algorithme de Berlekamp-Massey. Cet algorithme s'applique au problème du décodage des codes géométriques C_Ω, pour lequel nous construirons une ≪ équation clé ≫.

3.3 Décodage des des codes géométriques C_Ω

3.3.1 Préliminaires

La présentation de l'algorithme de décodage présenté ici est inspirée de [35]. Cet algorithme est valide pour les codes à un point $C_\Omega(D, rQ)$ où Q est la place à l'infini d'une courbe en position spéciale. On a donc une courbe affine $\mathcal{C}_a$ dans l'espace affine de dimension m, telle que $\mathcal{C}$ est la clôture projective de $\mathcal{C}_a$ avec un unique point Q à l'infini, et soit $\mathcal{P} = \{P_1, \dots, P_n\}$ les autres points rationnels de $\mathcal{C}$, qui sont tous affines.

On définit l'espace $L(\infty Q)$ est l'ensemble des fonctions qui admettent un pôle seulement en Q, d'ordre quelconque. Soit $x_1, \dots, x_m$ les fonctions coordonnées de $\mathbb{A}^m(\mathbb{F}_q)$, et $o_1, \dots, o_m$ les ordres (pris positifs) en Q de $x_1, \dots, x_m$. On considère $\mathbb{F}_q[x_1, \dots, x_m]$, et pour un multi-indice $\mathbf{s} = (s_1, \dots, s_m)$, nous notons $\mathbf{x^s}$ le monôme $x_1^{s_1} \cdots x_m^{s_m}$. L'anneau $\mathbb{F}_q[x_1, \dots, x_m]$ est muni d'un ordre monomial $\leqslant$ défini par le degré pondéré d'un monôme :

$$\mathrm{wdeg}(\mathbf{x^s}) = \mathrm{wdeg}(x_1^{s_1} \cdots x_m^{s_m}) = o_1s_1 + \cdots + o_ms_m, \tag{3.5}$$

raffiné par l'ordre lexicographique $x_1 < \dots, x_m$. Nous notons aussi $\mathrm{wdeg}(\mathbf{s}) = \mathrm{wdeg}(\mathbf{x^s})$.

Dans ce contexte, on construit la somme formelle $D = P_1 + \cdots + P_n$, et on va décoder le code $C = C_\Omega(D, rQ) = C_L(D, rQ)^\perp$. On rappelle que

$$C_L(D, rQ) = \{(f(P_1), \dots, f(P_n)) \,;\, f \in L(rQ)\}$$

Une famille génératrice de $L(rQ)$ est l'ensemble des monômes $\mathbf{x^s}$ tels que $\mathbf{x^s} \in L(rQ)$. Une famille génératrice de $C_L(D, P)$ est donc

$$\{\mathrm{ev}(\mathbf{s});\ \mathbf{s} \in L(rQ)\} = \{(x^{\mathbf{s}}(P_1), \dots, x^{\mathbf{s}}(P_n));\ \mathbf{x^s} \in L(rQ)\}.$$

Algorithme 1 Itération de l'algorithme de Berlekamp-Massey. On décrit comment passer de la récurrence satisfaite par la séquence $S_1, \ldots, S_{j-1}$ à la récurrence satisfaite par la séquence $S_1, \ldots, S_j$. Le programme ci-dessous doit donc être encapsulé dans une boucle supérieure

Entrée : $S_1, \ldots, S_j$
σ le polynôme de récurrence valide jusqu'à l'étape $j-1$
σ_0 le dernier polynôme de récurrence ayant échoué avant l'étape $j-1$.
δ_0 la discrépance à l'endroit où σ_0 a échoué.
l : le nombre de fois où le polynôme courant σ n'a pas changé de degré.
L le degré de σ.

Sortie :
σ^+ : le polynôme de récurrence valide jusqu'à l'étape j
σ_0^+ : le dernier polynôme de récurrence ayant échoué avant l'étape j.
δ_0^+ la discrépance où σ_0^+ a échoué.
l^+ : le nombre de fois où le polynôme σ^+ n'a pas changé de degré.
L^+ le degré de σ.

Corps de la boucle :

$\overline{S_j} \leftarrow S_j + \sum \sigma_i S_{j-i}$
if $\overline{S_j} = S_j$ **then**
  $l = l + 1$
else
  if $2L \geqslant j$ **then**
    $\sigma^+ \leftarrow \sigma - \delta\delta_0^{-1} x^l \sigma_0$
    $l \leftarrow l + 1$
  else
    $\sigma^+ \leftarrow \sigma - \delta\delta_0^{-1} x^l \sigma_0$
    $\sigma_0^+ \leftarrow \sigma$
    $L^+ \leftarrow j - L$//Ligne magique de l'algorithme de Berlekamp-Massey
    $\delta_0^+ \leftarrow \delta$
    $l^+ \leftarrow 1$
  end if
end if

Donc le code $C_\Omega(D, rQ) = C_L(D, rQ)^\perp$ est l'ensemble des mots $c = (c_i)_{i=1,\dots,n}$ tels que

$$\sum_{i=1}^{n} c_i x^{\mathbf{s}}(P_i) = 0; \quad \forall \mathbf{s} \text{ tel que } \operatorname{wdeg}(\mathbf{s}) \leqslant r.$$

Le mot reçu est $y = (y_i)_{i=1,\dots,n}$, avec $y = c + e$, où $c = (c_i)_{i=1,\dots,n}$ est le mot de code transmis à recouvrir (inconnu) et $e = (e_i)_{i=1,\dots,n}$ est l'erreur (inconnue).

3.3.2 Transformée généralisée

Définition 18. *Soit $\mathcal{P} = \{P_1, \dots, P_n\} \subset \mathbb{F}_q^m$, la* transformée généralisée *sur $\mathbb{F}_q^n$ est la fonction*

$$\begin{array}{rccc} \mathcal{F}: & \mathbb{F}_q^n & \to & \mathbb{F}_q^{\mathbb{N}^m} \\ & e & \mapsto & E = (E_{\mathbf{s}})_{\mathbf{s}\in\mathbb{N}^m} \end{array}$$

où

$$E_{\mathbf{s}} = \sum_{i=1}^{n} e_i \mathbf{x}^{\mathbf{s}}(P_i), \quad \mathbf{s} \in \mathbb{N}^m. \tag{3.6}$$

On note que le tableau $E = (E_{\mathbf{s}})$ de la transformée est multi-dimensionnel et infini. Mais, dans le contexte du codage, nous sommes sur $\mathbb{F}_q$, les polynômes $x_i^q - x_i$ s'annule sur les points $P \in \mathcal{P}$, et donc le tableau E est cyclique dans chacune de ses dimensions. On remarque de plus que si $m = 1$, et $(P_1, \dots, P_n) = (\alpha^0, \alpha^1, \dots, \alpha^{n-1})$, on retrouve la transformée de Fourier classique.

Définition 19. *Soit I un idéal de $\mathbb{F}_q[x_1, \dots, x_m]$. Pour un ordre monomial donné, l'*ensemble delta *de I est l'ensemble de monômes qui ne sont pas le monôme de tête d'un polynôme de I.*

Définition 20. *Soit $P_1, \dots, P_n \in \mathbb{F}_q^m$, et I l'idéal s'annulant en les points de $P_1, \dots, P_n$, et Δ l'ensemble delta de I. La* transformée propre *de $e \in \mathbb{F}_q^n$ est la restriction $E_{|\Delta} = \{E_{\mathbf{s}}, \mathbf{s} \in \Delta\}$ à Δ de la transformée E définie en (3.6).*

Proposition 11. *La transformée $e \mapsto E$ est injective. La transformée $e \mapsto E_{|\Delta}$ est bijective.*

Démonstration. Prouvons que $e \mapsto E_{|\Delta}$ est injective. Considérons P_i, pour $i \in \{1, \dots, n\}$. Soit e tel que $E_{|\Delta}$ est nul. Soit $A(\mathbf{x}) \in \mathbb{F}_q[\mathbf{x}]$ qui s'annule en tout p_j, $j \neq i$, et tel que $A(P_i) \neq 0$. Soit $A'(\mathbf{x}) = \sum A'_{\mathbf{r}} \mathbf{x}^{\mathbf{r}}$ la forme normale de f relativement à I, qui ne contient que des monômes $\mathbf{x}^{\mathbf{r}} \in \Delta$, nous avons

$$\begin{aligned} 0 &= \sum_{\mathbf{r}\in\Delta} A'_{\mathbf{r}} E_{\mathbf{r}} \\ &= \sum_{\mathbf{r}\in\Delta} A'_{\mathbf{r}} \sum_{i=1}^{n} e_i \mathbf{x}^{\mathbf{r}}(P_i) \\ &= \sum_{i=1}^{n} e_i A'(P_i) \\ &= e_i A(P_i). \end{aligned}$$

Donc $e_i = 0$, et ce raisonnement tient pour tout $i \in \{1, \dots, n\}$. La surjectivité provient du fait que $\mathbb{F}_q^n$ et l'espace des $E_{|\Delta}$ sont de même dimension. □

Rappelons que dans la situation du décodage, on a $y = c + e$, où c est le mot de code, e est l'erreur et y est le mot reçu. Comme $c \in C_\Omega(D, rQ) = C_l(D, rQ)^\perp$, on a, pour $\mathbf{s}$ tel que $\mathrm{wdeg}(\mathbf{s}) \leq r$:

$$\sum_{i=1}^{n} y_i \mathbf{x}^{\mathbf{s}}(P_i) = \sum_{i=1}^{n} c_i \mathbf{x}^{\mathbf{s}}(P_i) + \sum_{i=1}^{n} e_i x^{\mathbf{s}}(P_i) = \sum_{i=1}^{n} e_i \mathbf{x}^{\mathbf{s}}(P_i).$$

Donc la connaissance de y (le mot reçu), donne la connaissance des syndromes

$$S_{\mathbf{s}} = \sum_{i=1}^{n} e_i \mathbf{x}^{\mathbf{s}}(P_i), \quad \mathrm{wdeg}(\mathbf{s}) \leq r.$$

Définition 21. *Soit $\mathcal{C}$ une courbe en position spéciale, soit $C_\Omega(D, rQ)$ le code géométrique à un point sur cette courbe. Soit y le mot reçu, les* syndromes *de l'erreur e associée à y sont les $E_{\mathbf{s}}$*

$$E_{\mathbf{s}} = \sum_{P \in \mathcal{P}} y_P x^{\mathbf{s}}(P), \quad \mathrm{wdeg}(\mathbf{s}) \leqslant r.$$

Notons $(E_{\mathbf{s}})_{\mathbf{s} \in \Delta}$ la transformée de e. Comme dans le cas des codes de Reed-Solomon, le décodeur est partiellement aveugle : il ne connait qu'une partie de la transformée de e : les syndromes. Mais si la distance minimale est $2t+1$, alors il est garanti que pour un ensemble de syndromes $E_{\mathbf{s}}$, $\mathrm{wdeg}(\mathbf{s}) \leqslant r$, il existe au plus une erreur de poids t admettant ce jeu de syndromes.

Exemple 5. *Soit la courbe hermitienne $H \subset \mathbb{F}_{16}^2$, définie par l'équation $x^5 - y^4 - y = 0$ sur $\mathbb{F}_{16}$. Elle admet un seul point à l'infini, et soit $\mathcal{P}$ l'ensemble des 64 points $\mathbb{F}_{16}$-rationnels. On a que x est d'ordre -4 en l'infini, et y d'ordre -5, de qui nous donne l'ordre monomial défini par $4i + 5j$ rafiné par l'ordre lexicographique $x < y$. Une base de Gröbner de $\mathcal{P}$ est*

$$\{y^4 - x^5 + y, x^{16} - x\},$$

et l'ensemble Δ est $\Delta = \{(i, j);\ 0 \leqslant i < 16,\ 0 \leqslant j < 4\}$. On construit le code géométrique $[64, 46, 13]_{64}$ qui est l'orthogonal de $C_L(D, 23Q)$. Soit e une error de poids 6 dont les coefficients non nuls, indexés par les points, sont exactement

$$\begin{aligned} e_{(1,\alpha^8)} &= \alpha^{11}, \\ e_{(\alpha^2,\alpha^{12})} &= \alpha^{14}, \\ e_{(\alpha^3,\alpha^2)} &= \alpha^2, \\ e_{(\alpha^{11},\alpha^3)} &= \alpha, \\ e_{(\alpha^{12},\alpha^4)} &= \alpha^{14}, \\ e_{(\alpha^{14},\alpha^{11})} &= \alpha^3. \end{aligned}$$

Le tableau des syndrômes connus de e, $E = (E_{ij})_{4i+5j \leqslant 23}$ est présenté dans la table 3.1 La transformée propre $w \mapsto E_{|\Delta}$ est un isomorphisme d'espaces vectoriels de dimension 64 sur $\mathcal{F}^{16}$.

x^i \ y^j	0	1	2	3	4	5	6	...
0	0	α^6	1	0	α^5	$\star$	$\star$	...
1	α^{12}	α^3	α^4	α^6	$\star$	$\star$	...	
2	1	α^7	1	α^2	$\star$	$\star$	...	
3	α	α^8	α^5	$\star$	$\star$	...		
4	0	1	$\star$	$\star$	...			
5	α^9	$\star$	...					
6	$\star$							
$\vdots$	$\vdots$							

TABLE 3.1 – Un exemple de tableau des syndrômes. Chaque (i, j) correspond à un monôme $x^i y^j$, et les $\star$ sont des « syndromes inconnus ». La courbe est la courbe hermitienne $y^5 - x^4 - x = 0$ sur $\mathbb{F}_{16}$, et on considère les monômes $x^i y^j$ tels que $5i + 4j \leq 23$.

Supposons que le mot de code c a été transmis, et que le mot $y = c + e$ a été reçu par le décodeur. Le décodeur essaye de retrouver e en essayant de retrouver les localisations des erreurs, c'est-à-dire l'ensemble

$$\mathrm{supp}(e) = \{i \in \{1, \ldots, n\};\ e_i \neq 0\}.$$

Par définition du code $C_\Omega(\mathcal{P}, rQ)$, si C est la transformée de c, on a $C_{\mathbf{s}} = 0$, pour $\mathrm{wdeg}(\mathbf{s}) \leq r$. On a donc, pour $\mathrm{wdeg}(\mathbf{s}) \leq r$, $E_{\mathbf{s}} = Y_{\mathbf{s}}$, où E et Y sont les transformées de e et de y respectivement.

Définition 22. *Soit*

$$f = f(\mathbf{x}) = \sum f_{\mathbf{s}} x^{\mathbf{s}} \in \mathbb{F}[\mathbf{x}],$$

un polynôme en $\mathbf{x} = (x_1, \ldots, x_m)$. *On dit que le tableau* m*-dimensionnel infini* $E = (E_{\mathbf{s}})_{\mathbf{s} \in \mathbb{N}^m}$ *vérifie la relation de récurrence* f *si*

$$\sum_{\mathbf{s}} f_{\mathbf{s}} E_{\mathbf{s}+\mathbf{r}} = 0, \quad \textit{pour tout } \mathbf{r} \geqslant 0. \tag{3.7}$$

On dit aussi que la relation de récurrence m *dimensionnelle représentée par le polynôme* $f \in \mathbb{F}[x_1, \ldots, x_m]$ *est* valide *pour le tableau* E.

Proposition 12. *Soit* e *une erreur de transformée* $(E_{\mathbf{s}})_{\mathbf{s} \in \mathbb{N}^m}$. *L'ensemble des relations de récurrence valides pour le tableau* E *est un idéal, noté* $V(E)$. *Ses zéros sont les points* P_i *tels que* $e_i \neq 0$.

Démonstration. On a l'identité, pour tout $\mathbf{r} \in \mathbb{N}^m$:

$$\sum_{\mathbf{s}} f_{\mathbf{s}} E_{\mathbf{s}+\mathbf{r}} = \sum_{\mathbf{s}} f_{\mathbf{s}} \sum_{i=1}^{n} e_i x^{\mathbf{s}+\mathbf{r}}(P_i), \tag{3.8}$$

$$= \sum_{i=1 | e_i \neq 0}^{n} e_i x^{\mathbf{r}}(P_i) \sum_{\mathbf{s}} f_{\mathbf{s}} \mathbf{x}^{\mathbf{s}}(P_i), \tag{3.9}$$

$$= \sum_{i=1 | e_i \neq 0}^{n} e_i x^{\mathbf{r}}(P_i) f(P_i). \tag{3.10}$$

Donc si $f(P_i) = 0$, pour i tel que $e_i \neq 0$, alors f vérifie (3.7). Réciproquement, supposons que f vérifie (3.7), alors soit $a = (a_i)_{i\in\{1,\dots,n\}}$ le mot de composantes $a_i = e_i f(P_i)$. La relation (3.10) entraîne

$$\sum_{i=1}^{n} a_i \mathbf{x}^{\mathbf{r}}(P_i) = 0 \quad \text{pour tout } \mathbf{r} \in \mathbb{N}^m,$$

c'est-à-dire que la transformée généralisée de a est nulle. Comme elle injective, on a donc $a = 0$. Donc $f(P_i) = 0$ chaque fois que $e_i \neq 0$. □

Dans le cas des codes de Reed-Solomon, le polynôme localisateur d'erreur définit une relation de récurrence satisfaite par le tableau des syndromes. Dans le contexte des codes $C_\Omega(D, rQ)$, on cherche à déterminer l'idéal localisateur d'erreur, dont les zéros sont les points correspondant aux positions non nulles de l'erreur. Cet idéal est en correspondance avec l'idéal des récurrences linéaires de la transformée généralisée de l'erreur d'après la proposition 12. De même que l'algorithme de Berlekamp-Massey cherche la plus petite récurrence satisfaite par une suite $S_1, \dots, S_{2t}$, l'algorithme de Berlekamp-Massey-Sakata essaye de trouver les polynômes minimaux valides pour le tableau des syndrômes. Cela revient à trouver une base de Gröbner de l'idéal localisateur des positions d'erreur, et c'est ce que fait l'algorithme introduit Sakata et dénommé sous le nom d'algorithme de Berlekamp-Massey-Sakata [37, 36].

3.4 Algorithme de Berlekamp-Massey-Sakata

3.4.1 Ensembles deltas

On a donc $\leqslant$ un ordre monomial sur $\mathbb{F}_q[\mathbf{x}] = \mathbb{F}_q[x_1, \dots, x_m]$, qui, dans le cas du décodage des codes géométriques, est celui induit par le code géométrique $C_\Omega(P, kQ)$, et la valuation au point Q. Relativement à cet ordre, on définit le terme de tête $\mathrm{lead}(f)$, le coefficient de tête $\mathrm{lc}(f)$ d'un polynôme f.

On note $|$ l'ordre partiel de divisibilité entre les monômes : on a pour $\mathbf{r} = (r_1, \dots, r_m)$, $\mathbf{s} = (s_1, \dots, s_m)$, $\mathbf{r}|\mathbf{s}$ si $r_i \leq s_i$, $1 \leqslant i \leqslant m$. Tout ordre monomial respecte l'ordre de divisibilité.

Définition 23. *Soit $\mathcal{F}$ un sous ensemble de $\mathbb{F}_q[\mathbf{x}]$, et un ordre monomial fixé. On définit*

$$\Delta(\mathcal{F}) = \{\mathbf{s} \in \mathbb{N}^m : \forall f \in \mathcal{F},\ \mathrm{lead}(f) \not\leqslant \mathbf{s}\}.$$

Définition 24. *Un ensemble Δ est un* ensemble delta *si, pour tout $\mathbf{r}, \mathbf{s} \in \mathbb{N}^m$, $\mathbf{s} \in \Delta$, $\mathbf{r} \leqslant \mathbf{s} \implies \mathbf{r} \in \Delta$.*

Définition 25. *Soit $\Delta \subset \mathbb{N}^m$. Un* coin intérieur *de Δ est un multi-indice $\mathbf{r} \in \Delta$ maximal pour la relation de divisibilité. Un* coin extérieur *de Δ est un multi-indice $\mathbf{s} \notin \Delta$ minimal pour la relation de divisibilité.*

On note $\mathrm{Int}\,\Delta$ l'ensemble des points intérieurs, et $\mathrm{Ext}\,\Delta$ l'ensemble des points extérieurs de Δ.

Un ensemble delta est complètement déterminé par ses coins extérieurs de la manière suivante (Fig.3.1) :

$$\Delta = \{\mathbf{r} : \forall \mathbf{u} \in \mathrm{Ext}\,\Delta\ \mathbf{u} \not\leqslant \mathbf{r}\}.$$

.

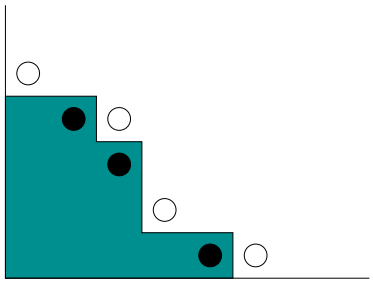

FIGURE 3.1 – Ensemble delta, coins intérieurs, coins extérieurs

3.4.2 Polynômes valides et invalides

Proposition 13. *Soit E un tableau m-dimensionnel, et f valide pour le tableau E. Pour un ordre monomial donné, et un indice du tableau $\mathbf{u}$ tel que* $\mathrm{lead}(f)|\mathbf{u}$*, la formule (3.7) donne*

$$E_{\mathbf{u}} = \frac{-1}{\mathrm{lc}(f)} \sum_{\mathbf{p}<\mathbf{u}} f_{\mathrm{lead}(f)-\mathbf{u}+\mathbf{p}} E_{\mathbf{p}}. \tag{3.11}$$

Démonstration. Une fois l'ordre monomial fixé, l'équation (3.7) (qui définit la validité) donne

$$\sum_{s \leqslant \mathrm{lead}(f)} f_{\mathbf{s}} E_{\mathbf{r}+\mathbf{s}} = 0.$$

Le terme le plus grand dans cette somme est $\mathbf{u} = \mathbf{r} + \mathrm{lead}(f)$, et on réécrit :

$$\sum_{s \leqslant \mathrm{lead}(f)} f_{\mathbf{s}} E_{\mathbf{u}-\mathrm{lead}(f)+\mathbf{s}} = 0,$$

avec $\mathrm{lead}(f)|\mathbf{u}$. Le changement d'indice $\mathbf{p} = \mathbf{u} - \mathrm{lead}(f) + \mathbf{s}$ permet d'obtenir :

$$\sum_{\mathbf{p} \leqslant \mathbf{u}} f_{-\mathbf{u}+\mathrm{lead}(f)+\mathbf{p}} E_{\mathbf{p}} = 0,$$

qui correspond bien (3.11) une fois le terme de tête isolé. □

Définition 26. *Un polynôme f est* invalide *à la position $\mathbf{u}$ si* $\mathrm{lead}(f)|\mathbf{u}$ *et si*

$$E_{\mathbf{u}} \neq \frac{-1}{\mathrm{lc}(f)} \sum_{\mathbf{p}<\mathbf{u}} f_{\mathrm{lead}(f)-\mathbf{u}+\mathbf{p}} E_{\mathbf{p}}.$$

Sinon, f est valide *à la position $\mathbf{u}$.*

On note qu'un polynôme tel que $\mathrm{lead}(f) \not| \mathbf{u}$ est automatiquement valide à la position $\mathbf{u}$, et que cette notion dépend de l'ordre monomial choisi.

Proposition 14. *Si f est valide à la position $\mathbf{u}$, alors $x^{\mathbf{p}} f$ est valide à la position $\mathbf{u}$.*

Définition 27. *Un polynôme f est* valide jusqu'à la position $\mathbf{u}$ *s'il est valide à toute les positions $\mathbf{r} \leqslant \mathbf{u}$. On note $V_{\mathbf{u}}(E)$ l'ensemble de polynômes valides jusqu'à la position $\mathbf{u}$.*

Soit $\mathbf{u}$ un indice du tableau, et $\mathbf{u}^+$ l'indice suivant pour l'ordre monomial. On a alors les inclusions suivantes :

$$\begin{array}{ccccccc} \mathbf{0} & < & \mathbf{u} & < & \mathbf{u}^+ & < & \infty \\ \mathbb{F}_q[\mathbf{x}] & \supseteq & V_{\mathbf{u}}(E) & \supseteq & V_{\mathbf{u}^+}(E) & \supseteq & V(E) \\ \emptyset & \subseteq & \Delta(V_{\mathbf{u}}(E)) & \subseteq & \Delta(V_{\mathbf{u}} + (E)) & \subseteq & \Delta(V(E)) \end{array}$$

Nous allons d'abord donner une méthode pour passer d'un ensemble de polynômes valides jusqu'à la position $\mathbf{u}$ à un ensemble de positions valides jusqu'à la position $\mathbf{u}^+$, tout en construisant l'ensemble $\Delta(V_{\mathbf{u}^+}(E))$. Cette brique élémentaire sera ensuite encapsulée dans une boucle supérieure.

3.4.3 Prédictions

Définition 28. *Soit $\mathbf{u}$ un indice du tableau E, et f valide pour toutes les positions $\mathbf{r} \leqslant \mathbf{u}$. La prédiction de f à l'indice suivant $\mathbf{u}^+$ est :*

$$P_{\mathbf{u}^+}(f) = \frac{-1}{\mathrm{lc}(f)} \sum_{\mathbf{p}<\mathbf{u}^+} f_{\mathrm{lead}(f)-\mathbf{u}^++\mathbf{p}} E_{\mathbf{p}}.$$

C'est ce que vaudrait $E_{\mathbf{u}^+}$ si était f valide jusqu'à la position $\mathbf{u}^+$.

Proposition 15. *Soit f et g valides jusqu'à $\mathbf{u}$, et tels que* $\mathrm{lead}(f) + \mathrm{lead}(g)$ *divise $\mathbf{u}^+$. Alors les prédictions de f et de g en $\mathbf{u}^+$ sont les mêmes :*

$$P_{\mathbf{u}^+}(f) = P_{\mathbf{u}^+}(g).$$

Démonstration. On voir que l'expression suivante est symétrique en f et g.

$$\begin{aligned} P_{\mathbf{u}^+}(f) &= \frac{-1}{\mathrm{lc}(f)} \sum_{\mathbf{p}<\mathbf{u}^+} f_{\mathrm{lead}(f)-\mathbf{u}^++\mathbf{p}} E_{\mathbf{p}}. \\ &= \frac{-1}{\mathrm{lc}(f)} \sum_{\mathbf{s}<\mathrm{lead}(f)} f_{\mathbf{s}} E_{\mathbf{s}+\mathbf{u}^+-\mathrm{lead}(f)} \\ &= \frac{-1}{\mathrm{lc}(f)} \sum_{\mathbf{s}<\mathrm{lead}(f)} f_{\mathbf{s}} P_{\mathbf{s}+\mathbf{u}^+-\mathrm{lead}(f)}(g) \\ &= \frac{-1}{\mathrm{lc}(f)\,\mathrm{lc}(g)} \sum_{\mathbf{s}<\mathrm{lead}(f)} f_{\mathbf{s}} \sum_{\mathbf{r}<\mathrm{lead}(g)} g_r E_{\mathbf{r}+\mathbf{s}+\mathbf{u}^+-\mathrm{lead}(f)-\mathrm{lead}(g)} \\ &= \frac{-1}{\mathrm{lc}(f)\,\mathrm{lc}(g)} \sum_{\substack{\mathbf{s}<\mathrm{lead}(f) \\ \mathbf{r}<\mathrm{lead}(g)}} f_{\mathbf{s}} g_{\mathbf{r}} E_{\mathbf{r}+\mathbf{s}+\mathbf{u}^+-\mathrm{lead}(f)-\mathrm{lead}(g)}. \end{aligned}$$

□

Définition 29. *Soit g valide jusqu'à la position $\mathbf{u}$ et invalide en $\mathbf{u}^+$. La* portée (span) *de g est*

$$\mathrm{span}(g) = \mathbf{u}^+ - \mathrm{lead}(g).$$

Théorème 8. *Soit* $g \notin V_{\mathbf{u}^+}(E)$. *Alors* $\mathrm{span}(g) \in \Delta(V_{\mathbf{u}^+}(E))$.

Démonstration. Soit $\mathbf{r} = \mathrm{lead}(g) + \mathrm{span}(g)$, $\mathbf{r}$ est la première position où g est invalide. Alors g est valide jusqu'à la position $\mathbf{r}$ non incluse, avec $\mathbf{r} \leqslant \mathbf{u}^+$. Supposons qu'il existe un polynôme $f \in V_{\mathbf{u}^+}(E)$ tel que $\mathrm{lead}(f) = \mathrm{span}(g)$. Alors comme f et g sont tous deux valides jusqu'à la position $\mathbf{r}$ non incluse, on devrait avoir la même prédiction $P_{\mathbf{r}}(f) = P_{\mathbf{r}}(g)$, ce qui est impossible puisque f est valide en $\mathbf{r}$ alors que g ne l'est pas. □

Définition 30. *Soit* $g \in \mathbb{F}[\mathbf{x}] \setminus V(E)$. *On dit que* g *est un* témoin *de* $\mathrm{span}(g)$. *Soit* $\mathcal{G} \in \mathbb{F}[\mathbf{x}] \setminus V(E)$, *et* Δ *un ensemble delta. On dit que* $\mathcal{G}$ *est un* ensemble témoin *de* Δ, *si* $\mathcal{G}$ *contient un témoin de chaque sommet intérieur de* Δ.

Théorème 9. *Soit* $\mathcal{F} \subset V_{\mathbf{u}}(E)$, *tel qu'il existe* $\mathcal{G} \subset \mathbb{F}[\mathbf{x}] \setminus V_{\mathbf{u}}(E)$ *qui soit un ensemble témoin de* $\Delta(\mathcal{F})$. *Alors* $\Delta(\mathcal{F}) = \Delta(V_{\mathbf{u}}(E))$.

Démonstration. Puisque $\mathcal{F} \subset V_{\mathbf{u}}(E)$, on a $\Delta(V_{\mathbf{u}}(E)) \subset \Delta(\mathcal{F})$. D'autre part, comme $\mathcal{G}$ est témoin de $\Delta(\mathcal{F})$, $\Delta(\mathcal{G}) = \Delta(\mathcal{F})$. Or $\mathcal{G} \subset \mathbb{F}[\mathbf{x}] \setminus V_{\mathbf{u}}(E)$, donc $\Delta(\mathcal{G}) \subset \Delta(V_{\mathbf{u}}(E))$. □

Théorème 10. *Soit* $\mathcal{F} \subset V(E)$, *tel qu'il existe* $\mathcal{G} \subset \mathbb{F}[\mathbf{x}] \setminus V(E)$ *qui soit un ensemble témoin de* $\Delta(\mathcal{F})$. *Alors* $\Delta(\mathcal{F}) = \Delta(V(E)$, *ce qui entraîne que* $\mathcal{F}$ *est une base de Gröbner de* $V(E)$.

3.4.4 L'algorithme

Pour l'indice courant $\mathbf{u}$, l'algorithme maintient deux ensembles $\mathcal{F}$ et $\mathcal{G}$ de polynômes :

1. $\mathcal{F}$: un ensemble de polynômes valides jusqu'à la position $\mathbf{u}$: il y en un par élément extérieur de l'ensemble Δ de $V_{\mathbf{u}}(E)$;
2. $\mathcal{G}$: un ensemble de polynômes témoins de l'ensemble Δ de $V_{\mathbf{u}}(E)$.

Le principe est de combiner des polynômes valides jusqu'à $\mathbf{u}$, donc issus de $\mathcal{F}$, en les combinant à des polynômes précédemment invalides, donc issus de $\mathcal{G}$, pour passer de $\mathbf{u}$ à $\mathbf{u}^+$.

Pour passer de la position $\mathbf{u}$ à la position $\mathbf{u}^+$, on contruit d'abord l'ensemble Δ de $V_{\mathbf{u}^+}(E)$. Pour cela, on teste les prédictions $P_{\mathbf{u}^+}(f)$ des polynômes de $\mathcal{F}$, qui sont valides en $\mathbf{u}$. On ajoute ceux dont la prédiction échoue à l'ensemble $\mathcal{G}$ pour obtenir $\mathcal{G}^+$. Ainsi les portées des polynômes de $\mathcal{G}+$ forment un ensemble témoin de l'ensemble Δ de $V_{\mathbf{u}^+}(E)$, par le théorème 8.

Une fois l'ensemble Δ de $V_{\mathbf{u}^+}(E)$ construit, pour chaque sommet extérieur $\mathbf{s} \in \Delta(V_{\mathbf{u}^+}(E)))$, on va contruire un polynôme valide en $\mathbf{s}$. Il y a trois cas :

1. soit il y a un ancien polynôme valide en $\mathbf{u}$ qui est toujours valide en $\mathbf{u}^+$: on le recycle ;
2. soit $\mathbf{s}$ ne divise pas $\mathbf{u}^+$, alors on « décale » un polynôme f valide en $\mathbf{u}$ pour que son terme de tête soit égal à $\mathbf{s}$; en effet, n'importe quel polynôme dont le terme de tête ne divise pas $\mathbf{u}^+$ est valide en $\mathbf{u}^+$.
3. soit $\mathbf{s}$ divise $\mathbf{u}^+$: on a va mettre à jour un polynôme $f \in \mathcal{F}$ en le combinant avec un polynôme de $\mathcal{G}$ pour avoir une prédiction juste en $\mathbf{u}^+$.

On a besoin d'une sorte de S-polynôme, qui permet de faire la combinaison précédente de deux polynômes dont les prédictions sont fausses (« invalides ») pour obtenir un polynôme dont la prédiction est juste (« valide »).

Définition 31. *Soit f tel que* $\mathrm{lead}(f)|\mathbf{s}$*, et g tel que* $(\mathbf{u}^+ - \mathbf{s})|\,\mathrm{span}(g)$*, avec* $\mathbf{s}|\mathbf{u}^+$*, comme dans les lignes 11 et 12 de la description de l'algorithme. On note* $S_{\mathbf{s},\mathbf{u}^+}(f,g)$ *le polynôme*

$$S_{\mathbf{s},\mathbf{u}^+}(f,g) = \mathbf{x}^{\mathbf{q}} f - c \cdot x^{\mathbf{p}} g,$$

avec

$$\begin{aligned} \mathbf{p} &= \mathrm{span}(g) - \mathbf{u}^+ + \mathbf{s}, \\ \mathbf{q} &= \mathbf{s} - \mathrm{lead}(f), \end{aligned}$$

et où c est le coefficient

$$\frac{\mathrm{lc}(f)\,(E_{\mathbf{u}^+} - P_{\mathbf{u}^+}(f))}{\mathrm{lc}(g)\,(E_{\mathbf{u}^+} - P_{\mathbf{u}^+}(g))}.$$

L'algorithme 2 donne l'étape incrémentale pour passer d'un ensemble minimal de polynômes valides jusqu'à la position $\mathbf{u}$ à un ensemble minimal de polynômes valides jusqu'à la position $\mathbf{u}^+$.

3.5 Procédure de vote pour le décodage des codes géométriques

Dans le problème du décodage des codes géométriques simplifiés $C_\Omega(D, rQ)$, les seuls syndrômes « connus » sont les $E_{\mathbf{s}}$, pour $\mathrm{wdeg}\,\mathbf{s} \leq r$. On ne connait donc pas le tableau total $E_{\mathbf{s}}$, $s \in \mathbb{N}^m$ des syndrômes. Pour remèdier à ce problème, on va déterminer au fur et à mesure les syndrômes inconnus. On encapsule l'étape élémentaire de l'algorithme Berlekamp-Massey-Sakata dans l'algorithme 4.

La procédure, non détaillée ici, est basée sur le principe suivant. Supposons que le syndrome $E_{\mathbf{u}+}$ soit inconnu. Les polynômes valides f jusqu'à la position $E_{\mathbf{u}}$ font des prédictions $P_{\mathbf{u}^+}(f)$. Parmi ces prédictions, certaines sont justes, d'autres sont fausses, mais on peut montrer, et c'est toute la beauté de la chose, que la prédiction juste a la majorité relavite parmi toutes les prédictions [27, 28]. Ce beau résultat a été obtenu initialement par Feng et Rao [13], mais l'interfaçage avec l'algorithme de Berlekamp-Massey-Sakata a été obtenu par les auteurs cités précédemment.

La complexité de cet algorithme est quadratique en la taille maximale de l'ensemble Δ que l'on essaye de construire. Dans le cas des courbes hermitiennes, on obtient une complexité de $O(n^{7/3})$ [36].

3.6 Variantes et généralisations

L'algorithme Berlekamp-Massey-Sakata se généralise à une famille de l tableau m-dimensionnels, $E^{(i)}$, $i \in \{1, \ldots, M\}$, avec $E^{(i)} = \left(E^{(i)}_{\mathbf{s}}\right)_{\mathbf{s}\in\mathbb{N}^m}$. On

Algorithme 2 Principe de l'algorithme Berlekamp-Massey-Sakata. On décrit simplement comment passer d'un élément $\mathbf{u}$ du tableau à l'élément suivant $\mathbf{u}^+$. Le programme ci-dessous doit donc être encapsulé dans une boucle supérieure

Entrée :
E, $\mathbf{u}$, $\mathbf{u}^+$,
$\mathcal{F}$ ensemble minimal de $V_{\mathbf{u}}(E)$,
$\mathcal{G}$ ensemble témoin de $V_{\mathbf{u}}(E)$
Sortie :
$\mathcal{F}^+$ ensemble minimal de $V_{\mathbf{u}^+}(E)$,
$\mathcal{G}^+$ ensemble témoin de $V_{\mathbf{u}^+}(E)$.
Corps de la boucle :
$\mathcal{N} \leftarrow$ les éléments de $\mathcal{F}$ invalides en $\mathbf{u}^+$
$\mathcal{G}^+ \leftarrow \mathcal{G} \cup \mathcal{N}$
$\Delta^+ \leftarrow \mathrm{span}(\mathcal{G}^+)$
for $\mathbf{s} \in \mathrm{Ext}\,\Delta^+$ **do**
  if $\exists f \in \mathcal{F} \setminus \mathcal{N}$ tel que $\mathrm{lead}(f) = s$ **then**
    $h^{\mathbf{s}} \leftarrow f$
  else if $s \not| \mathbf{u}^+$ **then**
    Trouver $f \in \mathcal{N}$ avec $\mathrm{lead}(f) | \mathbf{s}$
    $h^{\mathbf{s}} \leftarrow x^{\mathbf{s}-\mathrm{lead}(f)} f$
  else
    Trouver $f \in \mathcal{N}$, tel que $\mathrm{lead}(f) \mid \mathbf{s}$
    Trouver $g \in \mathcal{G}$, tel que $(\mathbf{u}^+ - \mathbf{s}) \mid \mathrm{span}(g)$
    $h^{\mathbf{s}} \leftarrow S_{\mathbf{s},\mathbf{u}^+}(f, g)$
  end if
end for
$\mathcal{F}^+ = \{h^{\mathbf{s}}; \mathbf{s} \in \mathrm{Ext}\,\Delta^+\}$

Algorithme 4 Encapsulation de l'étape élémentaire de l'algorithme BMS, avec procédure de vote

Entrée : $E_{\mathbf{s}}$, pour $\mathrm{wdeg}\,\mathbf{s} \leq r$, le tableau des syndrômes connus.
Initalisations $\mathbf{u} = 0$, $\mathcal{F} = \{1\}$, $\mathcal{G} = \emptyset$
Répéter
if Le syndrôme $E_{\mathbf{u}^+}$ est inconnu **then**
  Utiliser la procédure de vote pour obtenir $E_{\mathbf{u}^+}$.
end if
Utiliser l'étape élémentaire Berlekamp-Massey-Sakata pour trouver $\mathcal{F}^+$, $\mathcal{G}^+$
$\mathcal{F} \leftarrow \mathcal{F}^+$
$\mathcal{G} \leftarrow \mathcal{G}^+$
$\mathbf{u} \leftarrow \mathbf{u}^+$
Jusqu'à ce que tous les syndrômes sont déterminés.

cherche alors des récurrences $f = (f_{\mathbf{s}})_{\mathbf{s}\in\mathbb{N}^m}$ communes à tous les tableaux, c'est-à-dire telles que

$$\sum_{\mathbf{s}} f_{\mathbf{s}} E^{(i)}_{\mathbf{s}+\mathbf{r}} = 0, \qquad \text{pour tout } \mathbf{r} \geqslant 0, i \in \{1, \ldots, M\}.$$

Il existe déjà une généralisation de l'algorithme de Berlekamp-Massey pour la construction d'une récurrence linéaire satisfaite simultanément par M suites linéaires [14]. Sakata [38] a construit cette même généralisation de son algorithme, dans le cas de M tableaux m-dimensionels.

Chapitre 4

Décodage par interpolation

Dans ce chapitre nous nous intéressons au décodage des codes précédents, codes de Reed-Solomon et codes géométriques, vu ici comme codes d'évaluation. Cela est possible grâce aux propriétés de dualité de ces codes : le dual d'un code de Reed-Solomon de longueur n (de support complet $(\alpha^0, \alpha^1, \ldots, \alpha^{n-1})$, où α est une racine n-ième de l'unité, est un code de Reed-Solomon de longueur n, et le dual d'un code géométrique de longueur n construit sur une courbe $\mathcal{C}$ est un code géométrique de longueur n construit sur la même courbe.

Cette présentation des codes permet d'obtenir toute une autre famille d'algorithmes de décodage, avec notamment l'algorithme de Berlekamp-Welch [50, 17], qui est une alternative au décodage par « équation clé ».

Mais surtout, cette famille permet d'augmenter de beaucoup le rayon de correction τ, par rapport au rayon usuel $t = \lfloor \frac{d-1}{2} \rfloor$, qui est obtenu par le décodage classique. Pour obtenir ces rayons de décodage élevés, il faut relâcher les spécifications du problème du décodage : comme le rayon de décodage augmente de t à τ erreurs, on demande à l'algorithme de retourner la *liste* des mots de codes à distance inférieure à τ du mot reçu. On parle alors de *décodage en liste*. On résout donc un *problème différent*.

La grande similarité (pour les courbes en position spéciale) entre les codes de Reed-Solomon et les codes géométriques permet d'étendre les algorithmes obtenus dans le cas des codes de Reed-Solomon de manière surprenamment facile aux codes géométriques.

Ces progrès sont dus à Sudan [45], Shokrollahi et Wasserman [43], Guruswami et Sudan [19].

4.1 Algorithme de Berlekamp-Welch

4.1.1 Cas des codes de Reed-Solomon

Soit $\mathbb{F}_q$ le corps fini de cardinal q, $n \leq q$, et $x_1, \ldots, x_n \in \mathbb{F}_q$ distincts. Nous avons la fonction d'évaluation associée à $(x_1, \ldots, x_n)$, qui est

$$\begin{array}{rccc} \mathrm{ev}: & \mathbb{F}_q[x] & \to & \mathbb{F}_q^n \\ & f & \mapsto & (f(x_1), \ldots, f(x_n)) \end{array}$$

Le code de Reed-Solomon, de longueur n, de support $(x_1, \ldots, x_n)$, de dimension k est l'ensemble de $\mathrm{ev}\, f$, $f \in \mathbb{F}_q[x]$, $\deg f < k$. La distance minimale est $d =$

$n-k+1$, et ce code peut corriger $t = \lfloor \frac{n-k}{2} \rfloor$ erreurs. Pour simplifier nous supposerons que $n-k$ est pair, et donc que $t = \frac{n-k}{2}$.

Soit $c = \text{ev}\, f$ le mot transmis, et $y = (y_1, \ldots, y_n)$ le mot reçu. On considére le problème du décodage résolu quand f aura été retrouvé à partir de y. Nous allons utiliser un autre polynôme, le *polynôme localisateur* E, dont les racines sont les x_i tels que $y \neq f(x_i)$:

$$E = E(X) = \prod_{i; f(x_i) \neq y_i} (X - x_i)$$

Alors, pour tout $i \in \{1, \ldots, n\}$ deux choses l'une : ou bien $f(x_i) = y_i$, ou bien $E(x_i) = 0.$, Ce qui se traduit par

$$f(x_i)E(x_i) = y_i E(x_i), \quad i \in \{1, \ldots, n\}. \tag{4.1}$$

Dans ce système à n équations, les inconnues sont les coefficients de f et les coefficients de E. C'est un système non linéaire. Nous le simplifions en introduisons le polynôme $F = f \cdot E$, qui est de degré inférieur à $t + k$. Nous obtenons donc les équations

$$F(x_i) = y_i E(x_i), \quad i \in \{1, \ldots, n\}. \tag{4.2}$$

Nous présentons l'algorithme 4.1.1, qui fonctionne car la simplification $F = f \cdot E$ introduite plus haut n'empêche en rien de retrouver f sous la forme $-F/E$.

Algorithme 5 Algorithme de Berlekamp-Welch

Constantes :
q, $n < q$, k t, tels que $n-k$ est pair, et $t = \frac{n-k}{2}$.
$(x_1, \ldots, x_n) \in \mathbb{F}_q$, tous distincts.

Entrée :
$y = (y_1, \ldots, y_n)$ le mot reçu.

Sortie : $f \in \mathbb{F}_q[x]$, $\deg f(x) < k$, $d(\text{ev}\, f, y) \leq t$.

Interpolation : Trouver $E, F \in F_q[x]$ tels que

1. $F(x_i) - y_i E(x_i) = 0$, $i \in \{1, \ldots, n\}$;
2. $\deg F < t + k$;
3. $\deg E \leq t$

Recherche de racines : Retourner $f = -\frac{F}{E}$.
Si $\deg f \geq k$ ou $d(\text{ev}\, f, y) > \tau$, déclarer un échec de décodage.

Proposition 16 (Preuve de l'algorithme de Berlekamp-Welch)**.** *Pour tout mot y reçu, il existe toujours des polynômes $E, F \in \mathbb{F}_q[X]$ vérifiant les conditions 1. 2. et 3. de l'algorithme de Berlekamp-Welch. De plus, si le nombre d'erreur est inférieur ou égal à t, alors le polynôme $f(x) = \frac{F(x)}{E(x)}$ est correct.*

Démonstration. Dans le système d'équations défini par les conditions 1. 2. et 3 de l'algorithme, le nombre d'inconnues est $t+k+t+1 = 2t+k+1 \geqslant n+1$, alors que le nombre d'équations est n, il y a donc au moins une solution non nulle au système. Maintenant soit $c = \text{ev}\, f$ le mot de code, E le polynôme localisateur

associé, et soit $E_1, F_1 \in \mathbb{F}_q[X]$ et $E_2, F_2 \in \mathbb{F}_q[X]$ deux couples de solutions de ce système. Comme on a

$$F_1(x_i) = y_i E_1(x_i) \text{ et } F_2(x_i) = y_i E_2(x_i), \quad i \in \{1, \dots, n\},$$

on déduit

$$E_2(x_i)F_1(x_i) - E_1(x_i)F_2(x_i) = 0 \quad i \in \{1, \dots, n\}.$$

Or le polynôme $E_2(x)F_1(x) - E_1(x)F_2(x)$ est de degré strictement inférieur à $k+t+t = n$. Comme il a n racines, il est identiquement nul. On a donc $\frac{F_1}{E_1} = \frac{F_2}{E_2}$. Comme on a en particulier la solution E, et $F = f \cdot E$, qui montre qu'on a bien $\frac{F_1}{E_1} = -f(x)$ pour toute solution $E_1(x), F_1(x)$ du problème d'interpolation. □

4.1.2 Généralisation aux codes géométriques C_L

Nous considérons encore une courbe affine $\mathcal{C}$ de genre g, en position spéciale comme dans le chapitre précédent, admettant un unique point à l'infini Q. Ainsi les fonctions de $\mathbb{F}_q(\mathcal{C})$ n'admettant qu'un pôle à l'infini, sont des polynômes de $\mathbb{F}_q[X_1, \dots, X_m]/I$ où I désigne l'idéal de la courbe. On considère les fonctions ayant un pôle d'ordre r en Q, et aucun pôle ailleurs. L'ordre au pôle d'une telle fonction est donnée par un degré pondéré $\text{wdeg}_{o_1,\dots,o_m}(i_1, \dots, i_m) = o_1 i_1 + \cdots + o_m i_m$, où les o_i sont les ordres au pôles des $x_i = X_i \bmod I$, $i \in \{1, \dots, n\}$.

Soit $\{P_1, \dots, P_n\} \in \mathcal{C}(\mathbb{F}_q)$ des points affines $\mathbb{F}_q$-rationnels de $\mathcal{C}$, et le code que nous décoderons est $C_L(\mathcal{P}, rQ)$, qui est l'ensemble

$$\{\text{ev } f = (f(P_1), \dots, f(P_n));\ f \in L(rQ)\}$$

des mots de longueur n correspondants aux vecteurs d'évaluation des fonctions de $L(rQ)$.

Soit $c = \text{ev } f$ le mot transmis, et soit $y = (y_1, \dots, y_n)$ le mot reçu, et $I = \{i \in \{1, \dots, n\}; f(P_i) \neq y_i\}$, les positions d'erreur.

De même que dans le cas des codes de Reed-Solomon, on cherche à construire une fonction localisatrice d'erreur, $E \in \mathbb{F}_q[\mathcal{C}]$, telle que $E(P_i) = 0$, uniquement pour $i \in I$. Pour déterminer son ordre au pôle, il suffit de considérer le diviseur $D = (t+g)Q - \sum_{i \in I} P_i$. Par le théorème de Riemann, on a

$$\dim L(D) \geqslant \deg D - g + 1 = 1.$$

Il existe donc $E \in \mathbb{F}_q[\mathcal{C}]$ tel que $E(P_i) = 0$ et $\text{wdeg}_{o_1,\dots,o_m}(E) \leq t + g$.

Pour tout $i \in \{1, \dots, n\}$ on a : $f(P_i) = y_i$, ou $E(P_i) = 0$, ce qui se traduit par

$$f(P_i)E(P_i) = y_i E(P_i), \quad i \in \{1, \dots, n\}. \tag{4.3}$$

De même que dans le cas de l'algorithme de Berlekamp-Welch pour les codes de Reed-Solomon, on cherche donc $Q(Y) = F + YE \in \mathbb{F}_q[\mathcal{C}][Y]$ tel que :

- $F(P_i) - y_i E(P_i) = 0$, $i \in \{1, \dots, n\}$,
- $\text{wdeg}(F) \leqslant t + g + r$,
- $\text{wdeg}(E) \leqslant t + g$.

Proposition 17. *Si le nombre d'erreurs t vérifie $t < \frac{d-g}{2}$, alors la fonction $f \in L(rQ)$ correspondant au mot de code à distance inférieure à t est $f = -\frac{E}{F}$.*

Démonstration. Soit f la fonction telle que $d(\text{ev}\, f, y) \leq \tau$. Si $t < \frac{d-g}{2}$, alors $2t \leqslant d - g = n - r - g$, soit $t + r + g < n - t$. Le polynôme $Q(f)$ admet un degré pondéré inférieur à $t + r + g < n - t$. D'autre part, comme $f(P_i) = y_i$ en $\geqslant n - t$ points P_i, on a que la fonction $Q(f)$ a au moins $n - t$ zéros. Elle a plus de zéros que son degré pondéré, elle est donc nulle. On a donc $F(P_i) + y_i E(P_i) = 0$, et $F(P_i) + f(P_i)E(P_i) = 0$, $i \in \{1, \ldots, n\}$. □

L'algorithme est présenté en Fig. 6. Notons que cet algorithme ne décode pas jusqu'à la vraie capacité de correction $\lfloor \frac{d-1}{2} \rfloor$, il y a une pénalité due au genre.

Algorithme 6 Algorithme de Berlekamp-Welch pour les codes géométriques

Constantes :
q, n, r $t \leq \frac{d-g}{2}$, $\mathcal{C}$ une courbe affine de genre g, en « position spéciale ».
$(P_1, \ldots, P_n) \in \mathcal{C}(\mathbb{F}_q)$, tous distincts.
$\text{wdeg}_{o_1,\ldots,o_m}$ le degré pondéré défini par le point à l'infini de Q.

Entrée :
$y = (y_1, \ldots, y_n)$ le mot reçu.

Sortie :
$f \in \mathbb{F}_q[\mathcal{C}]$, $f \in L(rQ)$, telle que $d(\text{ev}\, f, y) \leq t$

Interpolation : Trouver $E, F \in \mathbb{F}_q[\mathcal{C}]$ tels que

1. $F(P_i) - y_i E(P_i) = 0$, $i \in \{1, \ldots, n\}$;
2. $\text{wdeg}(F) < t + g + r$;
3. $\text{wdeg}(E) \leq t + g$

Recherche de racines : Retourner une bonne représentation de $f = -\frac{F}{E}$.
Si $d(\text{ev}\, f, y) > t$ ou $f \notin L(rQ)$, déclarer un échec.

4.2 Décodage en liste et borne de Johnson

Pour les codes de Reed-Solomon, l'algorithme de décodage de Sudan est un algorithme de décodage en liste, de rayon $\tau > t$ où $t = \lfloor \frac{d-1}{2} \rfloor = \frac{n-k}{2}$ (quand $n - k$ est impair) est le rayon classique de décodage. Cela a la signification suivante : soit y le mot reçu, l'algorithme doit retourner *tous* les mots c du code de Reed-Solomon, tels que $d(c, y) \leq \tau$. Il y a une contrainte donnée par la taille de la liste sur le rayon τ de correction : quand celui ci devient trop grand, on devrait retourner un trop grand nombre de mots, voire exponentiel, ce qui n'est pas souhaitable. Un tel compromis est exprimé bar la *borne de Johnson*.

Proposition 18 (Borne de Johnson)**.** *Soit C un code de longueur n et de distance minimale d. Soit $y \in \mathbb{F}_q^n$, et $B(y, \tau) \cap C$ l'ensemble des mots de codes à distance τ de y. Alors*

$$|B(y, \tau) \cap C| \leq \frac{n(d - \tau)}{\tau^2 - 2n\tau + dn}$$

pourvu que le dénominateur soit positif. Cette condition est vérifiée pour $\tau < n - \sqrt{n(n-d)}$.

Démonstration. Voir [25] pour une preuve, et un énoncé plus affiné, faisant notamment intervenir la taille de l'alphabet $\mathbb{F}_q$. □

Donc on sait que si le rayon τ de décodage reste borné par $n - \sqrt{n(n-d)}$, quantité souvent appelée le rayon de Johnson, la taille de la liste est polynomiale en n. Notons que ce n'est pas une condition nécessaire : il est possible qu'au delà du rayon de Johnson, le nombre de mots de codes à distance τ de tout mot $y \in \mathbb{F}_q^n$ reste polynomial, pour certains codes. Toutefois, il existe des codes (non explicites) tels que la taille de liste est exponentielle dès qu'on dépasse le rayon de Johnson [21].

La situation n'est pas claire en ce qui concerne les codes de Reed-Solomon : il existe des bornes inférieures sur τ, telles qu'au dessus de ces bornes, il existe des mots y admettant un nombre non polynomial de mots de codes à distance τ de y [29, 40]. Mais il reste un large intervalle entre la borne de Johnson et ces bornes.

4.3 Algorithme de Sudan

L'algorithme de Sudan est une extension de l'algorithme de Berlekamp-Welch : alors que l'on cherchaut un polynôme $Q = F - YE \in \mathbb{F}_q[X][Y]$ qui interpole le mot reçu ; dans l'algorithme de Sudan, on cherche Q de plus haut degré en¿ Y plusieurs racines $f \in \mathbb{F}_q[X]$ telles que $Q(X, f) = 0$. Nous verrons alors que l'on gagne en rayon de correction. L'algorithme est présenté d'abord en 7, la preuve de correction ensuite, et le rayon de correction maximal possible est ensuite étudié. Notons que la condition 2. de l'algorithme 7 est le pendant des conditions 2. et 3. de l'algorithme 4.1.1.

Algorithme 7 Algorithme de Sudan pour les codes de Reed-Solomon

Constantes :
n, k t, tels que $n - k$ est pair, et τ le rayon de correction.
$(x_1, \ldots, x_n) \in \mathbb{F}_q$, tous distincts.

Entrée :
$y = (y_1, \ldots, y_n)$ le mot reçu.

Sortie : tous les $f \in \mathbb{F}_q[X]$, $\deg f < k$, tels que $d(\mathrm{ev}\, f, y) \leq \tau$.

Interpolation : Trouver un polynôme $Q \neq 0 \in \mathbb{F}_q[X][Y]$ tels que

1. $Q(x_i, y_i) = 0$, $i \in \{1, \ldots, n\}$;
2. $\mathrm{wdeg}_{1,k-1} Q < n - \tau$.

Recherche de racines : Retourner les polynômes $f = f(X)$ tels que
- $Q(X, f(X)) = 0$;
- $\deg f < k$;
- $d(\mathrm{ev}\, f, y) \leq \tau$.

Proposition 19. *Soit y le mot reçu, et $Q \in \mathbb{F}_q[X, Y]$ le polynôme construit dans l'algorithme de Sudan 7. Alors si $c = \mathrm{ev}(f(x))$ est à distance τ de y, avec $\deg f(x) < k$, alors $Q(x, f(x)) = 0$.*

Démonstration. Comme $\deg f < k$, et $\mathrm{wdeg}\, Q < n - \tau$, on a

$$\deg Q(X, f(X)) < n - \tau.$$

Ensuite, puisque $Q(x_i, y_i) = 0$ pour tout i, chaque fois que $f(x_i) = y_i$, on a $Q(x_i, f(x_i)) = 0$. Comme $d(evf, y) \leq \tau$, on a $f(x_i) = y_i$ pour au moins $n - \tau$

éléments x_i du support. Le polynôme $Q(X, f(X))$ a plus de racines que son degré, il est identiquement nul. □

Il reste à déterminer la capacité de correction τ. L'argument logique pour affirmer la validité de l'algorithme de Sudan est d'établir une condition suffisante pour que le polynôme Q existe quelque soit le mot reçu. Cette condition nous donnera la capacité de correction τ.

Proposition 20. *L'algorithme de Sudan corrige jusqu'à* $\lfloor n - \sqrt{2(k-1)n} \rfloor$ *erreurs.*

Démonstration. Remarquons que le vecteur des coefficients du polynôme Q est solution d'un système linéaire à n équations, donnée par $Q(x_i, y_i) = 0$, $i \in \{1, \ldots, n\}$. Pour que quelque que soit le mot reçu, un polynôme Q non nul existe, nous écrivons la condition suffisante suivante : le système linéaire doit avoir plus d'inconnues que d'équations, soit

$$N_Q > n,$$

où N_Q est le nombre de termes de $Q(x, y)$. La condition $\mathrm{wdeg}_{1,k-1} Q$ nous permet de borner le nombre de termes de Q :

$$N_Q = \sum_{i=0}^{l} \left(n - \tau - (k-1)\right),$$

avec $l = \lfloor \frac{\tau}{k-1} \rfloor$. On montre que $N_q \geqslant \frac{(n-\tau)^2}{2(k-1)}$. La condition $\frac{(n-\tau)^2}{2(k-1)} > n$ est donc suffisante pour s'assurer de l'existence de Q quelque soit le mot reçu. Elle donne $\tau < n - \sqrt{2(k-1)n}$. □

En notant $R = \frac{k}{n} \approx \frac{k-1}{n}$, le taux de transmission, et $\delta = \frac{\tau}{n}$, le taux d'erreur relatif, on obtient les courbes tracées en Fig. 4.3. On voit qu'on décode plus d'erreurs que l'algorithme classique lorsque R est petit, et aussi qu'on décode un nombre relatif d'erreurs proche de 1 lorsque le taux de transmission est proche de zéro. Par exemple, on peut décoder des taux d'erreurs de l'ordre de 99% (par exemple), alors que le décodage classique est limité à 50%. Notons toutefois que le rayon de décodage peut devenir négatif quand le taux de transmission est trop élevé.

4.4 Généralisation aux codes géométriques

Comme précédemment, l'algorithme se généralise aux codes géométriques. Ici, je me contenterai encore des courbes dites en position spéciale, avec un point à l'infini Q, en traduisant comme d'habitude la notion de degré par celle d'ordre au pôle Q. Soit donc $\mathcal{C}$ une courbe affine de genre g, admettant un unique point à l'infini, Q. Soit $P_1, \ldots, P_n$ n points de $\mathcal{C}(\mathbb{F}_q)$. Soit $r < n$, et soit $L(rQ)$ l'espace de Riemann-Roch associé à rQ. On note $D = P_1 + \cdots + P_n$.

On a, par le théorème de Riemann :

$$\dim L(rQ) \geqslant r - g + 1.$$

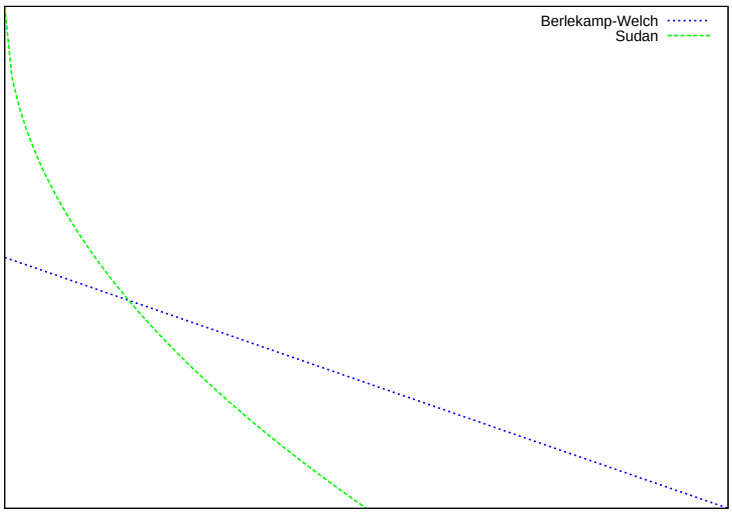

FIGURE 4.1 – Rayon $\delta = \frac{\tau}{n}$ de décodage de l'algorithme de Sudan

Le code géométrique $C = C_L(P1 + \cdots + P_n, rQ)$ est défini par la fonction d'évaluation :

$$\begin{array}{rccl} \text{ev}: & L(rQ) & \to & \mathbb{F}q^n \\ & f & \mapsto & \text{ev}\, f = (f(P_1), \ldots, f(P_n)) \end{array}$$

Décoder ce code revient à trouver toutes les fonctions de $f \in L(rQ)$ telles que $d(\text{ev}(f), y) \leq n - \tau$, où y est le mot reçu.

L'algorithme de Sudan pour les codes géométriques est simplement la traduction du cas des codes de Reed-Solomon, en écrivant le notion de degré en termes d'espaces associés à des diviseurs de supports réduits au point Q (algorithme 8, la condition 2. remplace la condition 2. de l'algorithme 7).

Algorithme 8 Algorithme de Sudan pour les codes géométriques

Constantes :
$\mathcal{C}$ une courbe affine en position spéciale, de genre g,
$D = P_1 + \cdots + +_n$, avec $P_1, \ldots, P_n \in \mathcal{C}(\mathbb{F}_q)$,
r un entier définissant le code $C_L(\mathcal{P}, rQ)$.

Entrée :
$y = (y_1, \ldots, y_n)$ le mot reçu.

Sortie : Tous les $f \in L(rQ)$, tels que $d(\text{ev}\, f, y) < \tau$.

Interpolation : Trouver $Q = \sum_{i=0}^{l} Q_i Y^i \in \mathbb{F}_q[C][Y]$ tel que

1. $Q \neq 0$;
2. $Q_i \in L((n - \tau - 1 - ri)Q)$;
3. $Q(y_i)(P_i) = 0$, pour $i \in \{1, \ldots, n\}$.

Recherche de racines : : retourner les $f \in L(rQ)$ tels que $Q(f) = 0$, et $d(\text{ev}\, f, y) \leq \tau$.

Proposition 21. *Soit $f \in L(rQ)$, tel que $d(\mathrm{ev}\, f, y) \leq \tau$, alors $Q(f) = 0$.*

Démonstration. Par construction du polynôme, $Q(f) \in L((n-\tau-1)Q)$. D'un autre coté, comme $d(\mathrm{ev}\, f, y) \leq \tau$, on a $f(P_i) = y_i$ en au moins $\mu \geq n - \tau$ points $P_{i_1}, \ldots, P_{i_\mu}$. Donc

$$Q(f) \in L((n-\tau-1)\, Q - (P_{i_1} + \cdots + P_{i_\mu})),$$

Or $\mu \geqslant n - \tau$, donc le degré du diviseur $(n-\tau-1)Q - (P_{i_1} + \cdots + P_{i_\mu})$ est négatif. L'espace associé est de dimension nulle : $Q(f) = 0$. □

Proposition 22. *Pour que pour chaque mot reçu, il existe un polynôme $Q \in \mathbb{F}_q[C][Y]$ vérifiant les conditions d'interpolations de l'algorithme 8, il suffit que*

$$\tau < n - \sqrt{2rn} - g. \tag{4.4}$$

Démonstration. L'espace des polynômes $Q = \sum_{i=0}^{l} Q_i Y^i$ tels que $Q_i \in L((n - \tau - 1 - ri)Q)$ est de dimension $\dim_Q$ minorée comme suit :

$$\begin{aligned}
\dim_Q &= \sum_{i=0}^{l} \dim L((n-\tau-1-ri)Q) \\
&\geqslant \sum_{i=0}^{l} (n-\tau-1-ri-g+1) \\
&= (l+1)(n-\tau-g) - r\frac{l(l+1)}{2} \\
&= (l+1)\left(n-\tau-g-r\frac{l}{2}\right).
\end{aligned}$$

On prend l tel que l est maximal vérifiant $rl < n - \tau - g$, car pour l supérieur à ce seuil, $\dim L((n-\tau-1-r)Q) < 0$. Donc

$$\begin{aligned}
\dim_Q &\geqslant (l+1)\left(n-\tau-g-\frac{n-\tau-g}{2}\right) \\
&= (l+1)\frac{n-\tau-g}{2} \\
&\geqslant \frac{(n-\tau-g)^2}{2r}
\end{aligned}$$

Si la dimension de cette espace est supérieure au nombre d'équations n, on est assuré d'avoir une solution nun nulle pour les vecteurs des coefficients de Q. On écrit donc $\dim_Q > n$, ce qui donne $\tau < n - \sqrt{2kn} - g$. □

On a encore une pénalité due au genre de la courbe, comme dans le cas de l'algorithme de Berlekamp-Welch.

4.5 Algorithme de Guruswami-Sudan

L'algorithme de Guruswami-Sudan est une amélioration radicale de l'algorithme de Sudan. Il permet de décoder toujours plus que le rayon de décodage

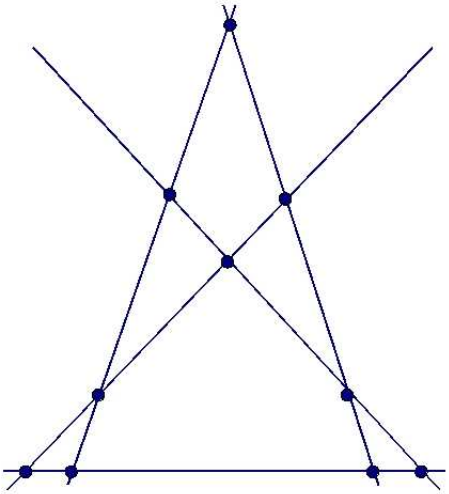

FIGURE 4.2 – Un exemple extrême de solutions au problème de décodage sur 10 points : par chaque point passent deux droites, et chaque droite est « à distance » 6 du mot reçu (elle passe par 4 points)
.

classique du décodage unique. En introduisant une interpolation avec « multiplicités » dans la condition 2. de l'algorithme 7.

La figure 4.5 montre un exemple forcé, qui montre pourquoi on a besoin d'introduire des multiplicités. On voit 10 points, avec des abscisses, $x_1, \ldots, x_{10}$, correspondant au support du code de Reed-Solomon, que l'on prend de dimension 2. C'est-à-dire que les polynômes que l'on évalue sont des polynômes affines, et leurs graphes sont des droites. Les ordonnées des points correspondent aux composantes du mot reçu. On voit que chacune des 5 droites (défini par $y - f_j(x) = 0$, $j \in \{1, \ldots, 5\}$ passe par quatre points, et leurs évaluations sont donc « à distance 6 » du mot reçu.

Mais on observe de plus qu'il y a au moins deux droites qui passent par chaque point. Un polynôme interpolateur tel que $y - f_j(x) | Q(x, y)$, $j \in \{1, \ldots, 5\}$ sera tel qu'il aura une multiplicité 2 en chaque point (x_i, y_i) $i \in \{1, \ldots, 10\}$. L'idée de Guruswami et Sudan est exactement de changer les conditions d'interpolation, en introduisant ces notions de multiplicités.

L'algorithme est présenté dans l'algorithme 9. Il nous faut définir la notion de multiplicité, comme suit (définition qui reste valide pour les corps de caractéristique positive).

Définition 32. *On dit qu'un polynôme $Q \in \mathbb{F}_q[X, Y]$ a une multiplicité à s en $(0, 0)$, s'il ne présente pas de termes de degré strictement inférieur à s. On dit que le polynôme Q a une multiplicité supérieure à s en (a, b), si le polynôme $Q(X + a, Y + b)$ a un multiplicité supérieure à s en $(0, 0)$.*

Proposition 23. *Soit $c = \operatorname{ev} f$ tel que $d(c, y) \leqslant \tau$, et $\deg f < k$. Alors $Q(X, f(X)) = 0$.*

Algorithme 9 Algorithme de Guruswami-Sudan pour les codes de Reed-Solomon

Constantes :
n, k t, tels que $n-k$ est pair, et τ le rayon de correction.
$x_1, \ldots, x_n \in \mathbb{F}_q$, tous distincts.
s un ordre de multiplicité.

Entrée :
$y = (y_1, \ldots, y_n)$ le mot reçu.

Sortie : $f \in \mathbb{F}_q[X]$, $\deg f < k$, $d(\mathrm{ev}\, f, y) \leq \tau$.

Interpolation : Trouver un polynôme $Q \neq 0 \in \mathbb{F}_q[X, Y]$ tels que

1. $\mathrm{mult}(Q, (x_i, y_i)) \geqslant s$, $i \in \{1, \ldots, n\}$;
2. $\mathrm{wdeg}_{1,k-1}(Q) < s(n-\tau)$.

Recherche de racines : Retourner les polynômes $f \in \mathbb{F}_q[X]$ tels que
- $Q(X, f(X)) = 0$;
- $\deg f < k$;
- $d(\mathrm{ev}\, f, y) \leq \tau$.

Démonstration. Pour chaque indice $i \in \{1, \ldots, n\}$ tel que $f(x_i) = y_i$, on a $(X - x_i)^s | Q(X, f(X))$. Soit I l'ensemble des indices i tels que $f(x_i) = y_i$, alors $\prod_{i \in I}(X - x_i)^s | Q(X, f(X))$. Mais $|I| \geq n - \tau$, puisque $d(\mathrm{ev}\, f, y) \leq \tau$. Donc un polynôme de degré supérieur à $s(n-\tau)$ divise $Q(X, f(X))$. D'autre part, la condition $\mathrm{wdeg}\, 1, k-1 Q < s(n-\tau$ entraîne que le degré de $Q(X, f(X))$ est strictement inférieur à $s(n-\tau)$. On a donc $Q(X, f(X)) = 0$. □

Proposition 24. *L'algorithme de Guruswami-Sudan peut corriger jusqu'à* $n - \sqrt{(k-1)n}$ *erreurs, quand l'ordre* s *de multiplicité est assez élevé.*

Démonstration. De même que précédemment, nous écrivons que le système linéaire doit avoir plus d'inconnues que d'équations, soit

$$N_Q > \binom{s+1}{2} n,$$

où N_Q est le nombre de termes de Q, et où on observe que la condition de multiplicité à chaque point (x_i, y_i) donne $\binom{s+1}{2}$ équation, $i \in \{1, \ldots, n\}$. La condition $\mathrm{wdeg}_{1,k-1}(Q(x, y)) < s(n-\tau)$ nous permet de borner le nombre de termes de $Q(x, y)$:

$$N_Q \geqslant \frac{(s(n-\tau)-1)^2}{2(k-1)}.$$

La condition $N_Q > \frac{s(s+1)}{2} n$ permet finalement d'obtenir

$$\tau < n - \sqrt{(1 + \frac{1}{s})(k-1)n} - \frac{1}{s}.$$

Quand s est assez grand, on obtient $\tau < \lfloor n - \sqrt{(k-1)n} \rfloor$. □

Or, pour les codes de Reed-Solomon, $k - 1 = n - d$ (borne de Singleton). L'algorithme de Guruswami-Sudan permet de décoder les codes de Reed-Solomon jusqu'à leur rayon de Johnson $n - \sqrt{n(n-d}$, et on peut dire qu'il est la réalisation algorithmique de la borne de Johnson.

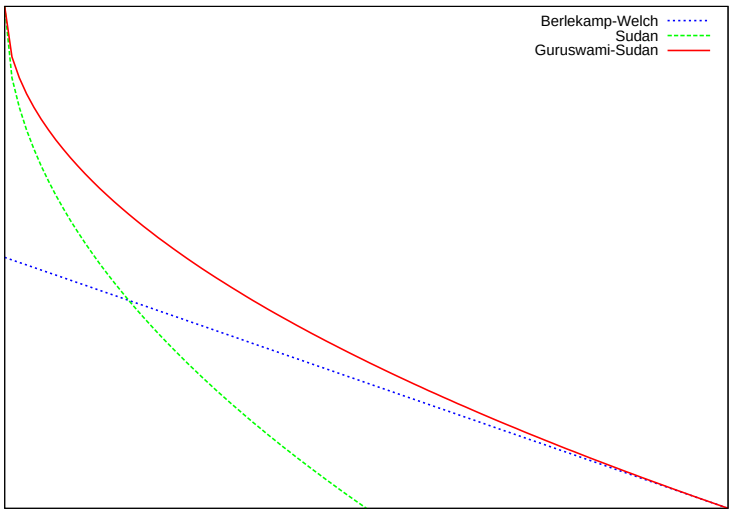

FIGURE 4.3 – Rayons $\delta = \frac{\tau}{n}$ de décodage des algorithmes de Sudan et de Guruswami-Sudan, en fonction du taux de transmission $R = \frac{k}{n}$

La figure 4.5 donne le rayon de décodage relatif $\frac{\tau}{n}$ en fonction du taux de transmission $R \approx \frac{k-1}{n}$, il vérifie $\tau \approx 1 - \sqrt{R}$. En terme de rayon relatif et de taux de transmission, la courbe $\delta = 1 - \sqrt{R}$ est toujours meilleure que la courbe $1 - \sqrt{2R}$, et aussi toujours meilleure que la courbe $\frac{1-R}{2}$ du rayon classique du décodage unique. On remarque aussi que l'algorithme de Guruswami-Sudan redonne bien le rayon de Sudan pour la multiplicité $s = 1$. De plus, la valeur minimale de s pour obtenir le rayon $\lfloor n - \sqrt{(k-1)n} \rfloor$ ne croît pas trop, de sorte que l'algorithme reste de complexité polynomial. Des formules explicites sont données dans [19].

Toutefois, même si la complexité reste polynomiale, il faut souvent prendre de grandes valeurs de s pour atteindre le rayon de décodage maximal, et en pratique, il est préférable souvent de décoder une erreur de moins, mais à un coût plus favorable [34].

4.6 Généralisation au codes géométriques

Encore une fois, cet exposé se limite aux courbes affines dites en position spéciale, avec un point à l'infini Q, en traduisant comme d'habitude la notion de degré par celle d'ordre au pôle Q. Soit donc $\mathcal{C}$ une courbe affine de genre g, admettant un unique point à l'infini, Q. Soit $P_1, \ldots, P_n$ n points de $\mathcal{C}(\mathbb{F}_q)$. Soit $r < n$, et soit $L(rQ)$ l'espace de Riemann-Roch associé à rQ. On note $D = P_1 + \cdots + P_n$, etle code que l'on cherche à décoder est $C_L(D, rQ)$.

Soit $y = (y_1, \ldots, Y - n)$ le mot reçu. L'algorithme 10 repose sur les même ingrédients : la recherche d'un polynôme $Q \in \mathbb{F}_q[C][Y]$, dont les coefficients sont de degré pondéré borné, qui passe par les points d'interpolation avec une certaine multiplicité.

La seule difficulté est de définir la multiplicité d'un polynôme $Q(y) \in \mathbb{F}_q[\mathcal{C}][Y]$

en un « point » (P_i, y_i), où $P_i \in \mathcal{C}(\mathbb{F}_q)$, et $y \in \mathbb{F}_q$. En développant relativement à y_j, on écrit

$$Q(y) = \sum_j (y - y_i)^j Q_{ij}, \quad Q_{ij} \in \mathbb{F}_q(\mathcal{C})[y].$$

et la multiplicité est alors définie par $\text{mult}(Q, (P_i, y_i)) = \min_j \{v_{P_i}(Q_{ij}) + j\}$.

Algorithme 10 Algorithme de Guruswami-Sudan pour les codes géométriques

Constantes :

n, r t, $\mathcal{C}$ une courbe affine de genre g, en position spéciale, de point Q à l'infini.
$P_1, \ldots, P_n \in \mathcal{C}$, tous distincts.
s un ordre de multiplicité.

Entrée :

$y = (y_1, \ldots, y_n)$ le mot reçu.

Sortie :

Interpolation : Trouver $Q(Y) = \sum_{i=0}^{l} Q_i Y^i \in \mathbb{F}_q[\mathcal{C}][Y]$ tel que

1. $Q \neq 0$;
2. $Q_i \in L((s(n-\tau) - 1 - ri)Q)$;
3. $\text{mult}(Q, ((P_i, y_i)) \geqslant s$, pour $i \in \{1, \ldots, n\}$.

Recherche de racines : : retourner les $f \in L(rQ)$ tels que $Q(f) = 0$, et $d(\text{ev}\, f, y) \leq \tau$.

Proposition 25. *Soit $Q(Y)$ comme dans l'algorithme 10, et $f \in L(rQ)$, alors $Q(f) \in L((s(n-\tau) - 1)Q)$.*

Démonstration. Pour chaque terme Q_i du polynôme $Q(Y)$, on a

$$\begin{aligned} -v_Q(Q_i f^i) &= -v_Q(Q_i) - i v_Q(f) \\ &\geq s(n-\tau) - 1 - ri + ir \\ &= s(n-\tau) - 1. \end{aligned}$$

□

Proposition 26. *Soit $f \in \mathbb{F}_q[\mathcal{C}]$ telle que $f(P_i) = y_i$, alors $v_{P_i}(Q(f)) \geqslant s$.*

Démonstration. Écrivons $Q(f) = \sum_j Q_{ij} f^j$, avec $Q_{ij} \in \mathbb{F}_q[\mathcal{C}]$. On a

$$\begin{aligned} v_{P_i}(Q(f)) &\geqslant \min_j v_{P_i}((f - y_i)^j G_{ij}) \\ &= \min_j \left(j v_{P_i}(f - y_i) + v_{P_i}(G_{ij})\right) \\ &\geqslant \min_j \left(j + v_{P_i}(G_{ij})\right) \\ &\geqslant s \end{aligned}$$

□

Proposition 27. *Soit $f \in L(rQ)$ telle que $d(ev(f), y)) \leq \tau$. Alors $Q(f) = 0$.*

Démonstration. Puisque $f(P_i) = y_i$ en au moins $n - \tau$ points $P_{i_1}, \ldots, P_{i_l}$, on a

$$Q(f) \in L(-s(P_{i_1} + \cdots + P_{i_\mu})).$$

Donc

$$Q(f) \in L((s(n-\tau)-1)\,Q - s(P_{i_1} + \cdots + P_{i_\mu})),$$

Or $\mu \geqslant n - \tau$, donc le degré du diviseur $s(n-\tau) - 1Q - s(P_{i_1} + \cdots + P_{i_\mu})$ est négatif. L'espace associé est nul : $Q(f) = 0$. □

Pour finir, comme d'habitude, il faut déterminer une condition suffisante pour le polynôme $Q(T)$ existe, indépendante du mot reçu. Pour cela il faut que la dimension de l'espace des coefficients du polynôme Q soit supérieur au nombre d'équations imposées par les conditions d'interpolation. Quand nous écrivons

$$Q(T) = \sum_i (Y - y_i)^j Q_{ij},$$

avec $v_{P_i} Q_{ij} \geqslant r - j$, chaque contrainte $v_{P_i}(Q_{ij}) \geqslant r - j$ impose $r - j$ équations linéaires sur les coefficients de Q. Chaque point donne donc un nombre d'équations

$$\sum_{j=0}^{r-1} (r - j) = \frac{r(r+1)}{2}.$$

Donc en comptant tous les points, nous avons $n\frac{r(r+1)}{2}$ équations sur les coefficients de Q. Nous devons donc avoir

$$\dim_Q > n\frac{r(r+1)}{2},$$

avec, en utilisant le même calcul que dans le cas de l'algorithme de Sudan simple :

$$\dim_Q > \frac{(s(n-\tau) - g)^2}{2r}.$$

En tirant τ, on obtient

$$\tau \leq n - \sqrt{rn(1 + \frac{1}{s})} - \frac{g}{s},$$

et quand s croît

$$\tau \leq n - \sqrt{rn}.$$

Or $r = n - d$, donc en fonction de la distance minimale le rayon τ vérifie $\tau \leq n - \sqrt{n(n-d)}$. Il est notable que dans le cas des codes de Reed-Solomon, et dans le cas des codes géométriques, le rayon de décodage a la même expression en fonction de la distance minimale, et que ce rayon est le rayon de Johnson énoncé dans la proposition 18. Toutefois, en fonction du taux de transmission $k = r - g + 1$, on obtient

$$\tau \leq n - \sqrt{(k + g - 1)n},$$

où on voit que le genre g de la courbe impacte négativement le rayon de décodage, par rapport aux codes de Reed-Solomon, correspondant au cas $g = 0$.

4.7 Retour de l'algorithme Berlekamp-Massey-Sakata

Soit $x_1, \ldots, x_n$ le support d'un code de Reed-Solomon de dimension k. En situation de décodage, soit $y = (y_1, \ldots, y_n)$ le mot reçu. Dans l'algorithme de Sudan, il faut trouver un polynôme $Q \in \mathbb{F}_q[X, Y]$, dont les coefficients sont de degrés pondéré $\mathrm{wdeg}_{1,k-1}$ petits, tel que

$$Q(x_l, y_l) = 0, \quad l \in \{1, \ldots, n\}.$$

C'est-à-dire

$$\sum q_{ij} x_l^i y_l^j = 0, \quad l \in \{1, \ldots, n\}.$$

On peut multiplier une telle équation par $x_i^{u'} y_i^{v'}$, pour n'importe quel couple (u', v'), pour obtenir de nouvelles équations. En notant $E^{(l)}$ le tableau $E^{(l)}_{u,v} = x_l^u y_l^v$, $u, v \geqslant 0$, on obtient le système d'équations

$$\sum q_{ij} E^{(l)}_{i+u',j+v'} = 0, \quad \forall u', v' \geq 0, \quad \forall l \in \{1, \ldots, n\}.$$

C'est-à-dire que le polynôme Q définit une relation de récurrence pour chaque tableau $E^{(l)}$, $l \in \{1, \ldots, n\}$.

Sakata propose l'astuce suivante : en faisant la somme des tableaux $E^{(l)}$, soit $E^{=} \sum E^{(l)}$, alors celui ci définit un système équivalent. On pourra donc utiliser l'algorithme de Berlekamp-Massey-Sakata présenté au chapitre précédent pour trouver les récurrences linéaires minimales pour le degré pondéré $\mathrm{wdeg}_{1,k-1}$.

Proposition 28. *Le polynôme $Q = \sum q_{ij} X^i Y^j \in \mathbb{F}_q[X, Y]$ définit une relation de récurrence pour chaque $E^{(l)}$ si et seulement si il définit une relation de récurrence pour E.*

Démonstration. Si le polynôme $Q(X, Y) = \sum q_{uv} X^u Y^v$ définit une relation de récurrence pour chaque $E^{(l)}$, alors par linéarité, il est une relation de récurrence pour E. Réciproquement, supposons que

$$S_{u',v'} := \sum q_{uv} E_{u+u',v+v'} = 0, \quad \forall u', v' \geq 0.$$

Alors on peut écrire

$$\begin{aligned} S &= \sum_{uv} q_{uv} \sum_{l=1}^{n} x_l^{u+u'} y_l^{v+v'} \\ &= \sum_{l=1}^{n} \sum_{uv} q_{uv} x_l^{u+u'} y_l^{v+v'} \end{aligned}$$

En substituant $u' = u' + u''$, on obtient

$$\begin{aligned} S_{u',v'} &= \sum_{l=1}^{n} x_l^{u'} \sum_{uv} q_{uv} x_l^{u+u''} y_l^{v+v'} \\ &= \sum_{l=1}^{n} x_l^{u'} S^{(l)}_{u'',v'} \end{aligned}$$

pour tout u'. Comme les x_l, $l \in \{1, \ldots, n\}$, sont distincts (c'est le support du code de Reed-Solomon), la matrice de Vandermonde $V(x_1, \ldots, x_n) = \left(x_l^{u'}\right)_{l,u'}$ associée est de déterminant non nul, on a donc $S^{(l)}_{u',v'} = 0$ pour $u', v' \geq 0$, $l \in \{1, \ldots, n\}$. □

Adaptation à l'algorithme de Guruswami-Sudan Dans l'algorithme de Guruswami-Sudan, utilisant la multiplicité s, on obtiendrait, pour tout $l \in \{1, \ldots, n\}$, et pour tout couple (i, j) tel que $i + j < s$, $n\binom{s+1}{2}$ tableaux $E^{(i,j,l)}$ dont le polynôme $Q(X, Y)$ est une relation de récurrence. La même astuce que précédemment, qui consiste à sommer sur tous les points du support, nous donne $\binom{s}{2}$ tableaux $E^{(i,j)}$ dont on cherche simultanément une relation de récurrence. Sakata [39, 38] a proposé une généralisation de son algorithme pour exactement résoudre ce problème là.

Chapitre 5

Conclusion

J'espère avoir introduit de manière motivé un partie du « zoo » des codes correcteurs d'erreur *algébriques*. Alors que la théorie algébrique des codes correcteurs d'erreur a culminé en 1995 avec le décodage des codes géométriques C_Ω a un coût raisonnable $O(n^{7/3})$ pour les codes définis sur les courbes hermitiennes, deux années plus tard, le décodage en liste a permis de franchir des barrières importantes en terme de capacité de correction. La problème de décoder au delà de $t = \lfloor \frac{d-1}{2} \rfloor$ résistait depuis trente ans, et seuls quelques résultats parcellaires permettaient d'augmenter le rayon de quelques unités.

J'ai aussi fait le choix de ne pas présenter les complexités des algorithmes considérés. Il y a deux raisons à cela. La première est d'abord que le problème peut-être considéré comme à moitié résolu quand on retombe sur un problème d'algèbre linéaire. Il est ensuite possible de réduire l'exposant de complexité en reconnaissant des matrices structurées, voire des formes polynomiales des équations, ce permet d'avoir des coûts quadratiques en la taille du problème (la longueur du code). La deuxième raison est que les algorithmes de décodage des codes sont en général implantés *in fine* de manière matériellle, et que les mesures classiques de complexité sont moins pertinentes.

Toutefois, on peut dire les choses suivantes. Le décodage classique, à base de syndrome, est beaucoup plus rapide que le décodage en liste, qui est encore trop lourd, bien que polynomial. Notamment l'algorithme de Berlekamp-Massey-Sakata a une complexité quadratique [37] en la taille du tableau dans lequel on cherche des relations de récurrence. Il y a de plus des propositions d'implantation pipelinées de l'algorithme de Berlekamp-Massey-Sakata [33] ou de son amélioration par Koetter [1, 30].

En ce qui concerne le décodage par interpolation à la Sudan, c'est l'étape d'interpolation qui est coûteuse, notamment quand le paramètre auxiliaire s de multiplicité augmente. Cela rend l'algorithme impraticable. Il y a alors, à mon sens, deux possibilités :

1. utiliser les meilleures méthodes du calcul formel, et faire le saut dans le monde des algorithmes rapides. Dans le cas des codes de Reed-Solomon, Brander obtient une complexité du type $\tilde{O}(l^5 n)$, où l est la taille maximale de la liste. Il est difficile de mesurer la pertinence de cette approche, car par exemple les codes de Reed-Solomon employés en pratique sont de longueur n petite, par exemple 255, pour conserver un petit alphabet (les octets). De plus les implantations visées sont matérielles. En revanche, les codes

géométriques sont par nature moins limités en longueur, et il semble plus pertinent d'utiliser des méthodes rapides, pour des codes longs. Pour les courbes hermitiennes, on obtient une complexité de l'ordre de $\tilde{O}(l5n^2)$. Ces résultats sont obtenus par Kristian Brander sans sa thèse. Il n'a toutefois pas réussi à implanter ses méthodes dans le cas des codes hermitiens, même en Magma.

2. transformer le problème à l'origine. Par exemple, Wu [51] fait un mélange de Berlekamp-Massey et de Guruswami-Sudan. L'algorithme de Berlekamp-Massey lui donne un résultat intermédiaire, qu'il met à profit pour réduire l'ordre s de multiplicité nécessaire. On peut aussi se poser la question de la sensibilité de l'algorithme à la donnée. L'algorithme de Guruswami-Sudan se comporte de manière uniforme et déterministe, ce qui ne semble pas très raisonnable. Une voie de recherche prometteuse serait de chercher un algorithme qui se comporte mieux quand l'erreur est petite.

Enfin, du point de vue théorique, Guruswami et Rudra ont encore progressé sur le rayon de décodage, en construisant des codes non standards (par exemple ce ne sont pas des codes linéaires), les codes de Reed-Solomon « repliés » [24], qui admettent un super rayon de décodage en liste. En fait ils atteignent la capacité du décodage en liste $\frac{\tau}{n} \approx 1 - \frac{k}{n} - \epsilon$, mais ces codes ont un alphabet qui croit très vite en fonction de la longueur, de manière exponentielle en $\frac{1}{\epsilon^2}$. Comme toujours les codes géométriques peuvent être utilisés en remplacement des codes de Reed-Solomon dans cette construction pour réduire la croissance de la taille de l'alphabet [23].

De manière plus réaliste, il reste le cas binaire, dont Guruswami fait un survol [22]. La question qui se pose est de savoir si on peut décoder un taux d'erreur $\frac{\tau}{n} = \frac{1}{2} - \epsilon$ avec des codes de taux $n = O(\frac{k}{\epsilon^2})$ constructibles en temps polynomial en un algorithme polynomial en n et $\frac{1}{\epsilon}$. On sait que des tels codes existent avec un tel rayon de décodage pour un tel rapport k/n (par un argument probabiliste), sans savoir les construire, ni bien sûr les décoder.

Annexe A

A.1 Preuve de l'algorithme de Berlekamp-Massey-Sakata

Théorème 11. *Il existe bien un poynôme f comme requis dans les étapes 8 et 11 de l'algorithme Berlekamp-Massey-Sakata.*

Démonstration. Il s'agit de montrer qu'il existe $f \in \mathcal{N}$ tel que $\text{lead}(f)|\mathbf{s}$. Le point $\mathbf{s}$ est un point extérieur de Δ^+, qui contient $\Delta(\mathcal{F})$.

Supposons d'abord que $\mathbf{s}$ est un point extérieur de $\Delta(\mathcal{F})$, alors il existe $f \in \mathcal{F}$ tel que $\text{lead}(f) = \mathbf{s}$. Comme nous sommes dans l'autre branche que celle du cas **then** du test 5 de l'algorithme, on a $f \in \mathcal{N}$.

On considère maintenant le cas où $\mathbf{s}$ n'est pas un coin extérieur de $\Delta(\mathcal{F})$. Le point $\mathbf{s}$ est néammoins un point de l'extérieur de $\Delta(\mathcal{F})$, et il existe $f(x) \in \mathcal{F}$ tel que $\text{lead}(f)$ soit un point extérieur de $\mathcal{F}$. De plus $\text{lead}(f) \in \Delta^+$. Comme $\Delta^+ \subseteq \Delta(V_{\mathbf{u}^+}(E))$, on a

$$\text{lead}(f) \in \Delta(V_{\mathbf{u}^+}(E)).$$

Mais aucun polynôme de terme de tête $\text{lead}(f)$ ne peut-être valide en $\mathbf{u}$. Donc $f \in \mathcal{N}$. □

Proposition 29. *Il existe bien un polynôme g comme requis dans l'étape 12 de l'algorithme BMS.*

Démonstration. Supposons que ce ne soit pas possible, comme $\mathcal{G}$ est l'ensemble témoin, cela entraîne que $\mathbf{u}^+ - \mathbf{s}$ est à l'extérieur de $\Delta(\mathcal{F})$, et il existe un coin extérieur $\mathbf{p}$ de $\Delta(\mathcal{F})$ tel que $\mathbf{p}|\mathbf{u}^+ - \mathbf{s}$. Il y a donc $f' \in \mathcal{F}$ tel que $\text{lead}(f') = \mathbf{p}$. On a donc

$$\text{lead}(f') + \text{lead}(f)|\mathbf{u}^+,$$

et par le théorème des prédictions, f et f' font la même prédiction en $\mathbf{u}^+$. Donc $f' \in \mathcal{N}$ et donc $\text{span}(f') \in \Delta^+$, i.e. $\mathbf{u}^+ - \mathbf{p} \in \Delta^+$. Comme $s|\mathbf{u}^+ - \mathbf{p}$, on a que $s \in \Delta^+$, ce qui est une contradiction. On a donc un g avec $\mathbf{u}^+ - \mathbf{s}| \text{span}(g)$. Notons que g est valide jusqu'à la position $\mathbf{u}$ non incluse, □

Proposition 30. *Les polynômes construits dans les étapes 6, 9 ou 13 sont de terme de tête s, et sont valides en $\mathbf{u}^+$.*

Démonstration. Le cas de l'étape 6 est clair. Dans le cas 9, $h^{\mathbf{s}}$ est de terme de tête $\mathbf{s}$, et ne peut être invalide en $\mathbf{u}^+$, puisqu'il ne peut pas fournir de prédiction ($\mathbf{s} \not/ \mathbf{u}^+$). D'autre part, comme f est valide jusqu'à $\mathbf{u}$, on a bien $f \in V_{\mathbf{u}^+}(E)$.

Montrons que le polynôme $h^{\mathbf{s}}$ construit en 13 admet $\mathbf{s}$ comme terme de tête. Rappelons la formule pour $S_{\mathbf{s},\mathbf{u}^+}(f,g)$:

$$S_{\mathbf{s},\mathbf{u}^+}(f,g) = x^{\mathbf{s}-\mathrm{lead}(f)} f - c x^{\mathrm{span}(g)-\mathbf{u}^+ + \mathbf{s}} g \tag{A.1}$$

où c est un coefficient annulateur bien choisi.

Le terme $x^{\mathbf{s}-\mathrm{lead}(f)} f$ admet $\mathbf{s}$ comme monôme de tête.

Soit $\mathrm{span}(g) = \mathbf{u}_0 - \mathrm{lead}(g)$, avec $\mathbf{u}_0 < \mathbf{u}+$, car g a bloqué précédemment. Le monôme de tête du deuxième terme de la soutraction est

$$\begin{aligned} \mathrm{span}(g) - \mathbf{u}^+ + \mathbf{s} + \mathrm{lead}(g) &= \mathbf{u}_0 - \mathrm{lead}(g) - \mathbf{u}^+ + \mathbf{s} + \mathrm{lead}(g) \\ &= \mathbf{u}_0 - \mathbf{u}^+ + \mathbf{s} \\ &< s. \end{aligned}$$

Donc la somme des deux termes admet bien $\mathbf{s}$ comme monôme de tête. Le fait que $S_{\mathbf{s},\mathbf{u}^+}(f,g)$ est valide en $\mathbf{u}^+$ se vérifie par le calcul. □

A.2 Obtention de l'algorithme de Berlekamp-Massey

Nous présentons comment l'algorithme de Berlekamp-Massey-Sakata redonne bien l'algorithme de Berlekamp-Massey, lorsque le tableau est unidimensionel. Nous avons alors les simplifications suivantes :

1. les monômes $\mathbf{u}$ sont des entiers ;
2. il n'y a qu'un seul ordre monomial, qui coïncide avec la relation de divisibilité ;
3. l'ensemble des récurrences minimales est un idéal principal représenté par un polynôme ;
4. l'ensemble témoin est le polynôme de plus haut degré qui n'est pas dans l'idéal ;
5. la portée $\mathrm{span}(g)$ d'un polynôme est la différence entre son degré et l'indice où il bloque : c'est le nombre de valeurs pendant lesquels il est valide.

Proposition 31. *Soit f^+ le polynôme produit à l'étape 9 de l'algorithme de Berlekamp-Massey présenté en Fig 11, dans le cas où le polynôme précécent f a échoué. On a $\deg f^+ = u + 1 - \deg f$. Dans l'autre cas, le degré est maintenu.*

Démonstration. Il suffit de calculer $S_{\mathbf{s},\mathbf{u}^+}(f,g) = \mathbf{x}^{\mathbf{q}} f - c \cdot x^{\mathbf{p}} g$, avec

$$\begin{aligned} \mathbf{p} &= \mathrm{span}(g) - \mathbf{u}^+ + \mathbf{s} \\ \mathbf{q} &= \mathbf{s} - \mathrm{lead}(f) \\ \mathbf{s} &= \mathbf{u}^+ - \deg(f) \end{aligned}$$

Soit $S_{\mathbf{s},\mathbf{u}^+}(f,g) = \mathbf{x}^{\mathbf{q}} f - c \cdot x^{\mathbf{p}} g = x^{\mathbf{s}-\mathrm{lead}(f)} f - c x^{\mathrm{span}(g)-\mathbf{u}^+ + \mathbf{s}} g$. Or $\mathrm{span}(g) = u - \deg(g)$, donc

$$\begin{aligned} \deg(x^{\mathrm{span}(g)-\mathbf{u}^+ + \mathbf{s}} g) &= \mathbf{u} - \deg(g) - (\mathbf{u}+1) + \mathbf{u} + 1 - \deg(f) + \deg(g) \\ &= \mathbf{u} - \deg(f), \end{aligned}$$

Algorithme 11 Algorithme de Berlekamp-Massey, dérivé de l'algorithme de Berlekamp-Massey-Sakata, à encapsuler dans une boucle supérieure

Entrée :

E, $\mathbf{u}$, $\mathbf{u}^+ = \mathbf{u} + 1$,
$\mathcal{F} = \{f\}$ ensemble minimal de $V_{\mathbf{u}}(E)$, réduit à f,
$\mathcal{G} = \{g\}$ ensemble témoin de $V_{\mathbf{u}}(E)$, réduit à g.

Sortie :

$\mathcal{F}^+ = \{f^+\}$ ensemble minimal de $V_{\mathbf{u}^+}(E)$,
$\mathcal{G}^+ = \{g^+\}$ ensemble témoin de $V_{\mathbf{u}^+}(E)$.

Corps de la boucle :

1: $\mathcal{N} \leftarrow$ les éléments de $\mathcal{F}$ invalides en $\mathbf{u}^+$, $\mathcal{N} = \{f\}$ ou $\emptyset$,
2: **if** $\mathcal{N} = \emptyset$ **then**
3: **continue**
4: **end if**
5: $\mathcal{G}^+ \leftarrow \{f\}$,
6: $\mathbf{s} = \mathbf{u}^+ - \deg f$ // Remplace 3 et 4 de BMS
7: // Le cas de la ligne 5 de BMS ne se produit pas
8: // Le cas de la ligne 7 de BMS ne se produit pas
9: $h^{\mathbf{s}} \leftarrow S_{\mathbf{s},\mathbf{u}^+}(f, g)$
10: $\mathcal{F}^+ = \{f^+\} = \{h^{\mathbf{s}}\}$

tandis que

$$\deg\left(x^{\mathbf{s}-\mathrm{lead}(f)} f\right) = \mathbf{s} = \mathbf{u}^+ - \deg(f) = \mathbf{u} + 1 - \deg(f).$$

Donc le terme de tête de $S_{\mathbf{s},\mathbf{u}^+}(f, g)$ a bien pour degré $\mathbf{u} + 1 - \deg(f)$.

□

Bibliographie

[1] J. B. Ashbrook, N. R. Shanbhag, R. Koetter, and R. E. Blahut. Implementation of a Hermitian decoder IC in 0.35 μm CMOS. In *Custom Integrated Circuits, 2001, IEEE Conference on.*, pages 297–300, 2001.

[2] E.F. Assmus and J.D. Key. Polynomial codes and finite geometries. In V.S. Pless, W.C. Huffman, and R.A. Brualdi, editors, *Handbook of coding theory*, volume II. North-Holland, 1998.

[3] Elwyn R. Berlekamp. *Algebraic coding theory.* McGraw-Hill, 1968.

[4] Elwyn R. Berlekamp, Robert J. McEliece, and Henk C. A. van Tilborg. On the inherent intractability of certain coding problems. *IEEE Transactions on Information Theory*, 24(3) :384–386, 1978.

[5] Richard E. Blahut. *Fast Algorithms for Digital Signal Processing.* Addison-Wesley, January 1985.

[6] Richard E. Blahut. *Algebraic Codes for Data Transmission.* Cambridge University Press, July 2002.

[7] J. Bruck and M. Naor. The hardness of decoding linear codes with preprocessing. *IEEE Transactions on Information Theory*, 36(2) :381–385, 1990.

[8] Qi Cheng. Hard problems of algebraic geometry codes. `http://arxiv.org/abs/cs.IT/0507026`, Jul 2005.

[9] Qi Cheng and Daqing Wan. Complexity of decoding positive-rate reed-solomon codes. In *ICALP '08 : Proceedings of the 35th international colloquium on Automata, Languages and Programming, Part I*, Lecture Notes in Computer Science, pages 283–293, Berlin, Heidelberg, 2008. Springer-Verlag.

[10] Chien-Yu Chien and I. M. Duursma. Geometric Reed-Solomon codes of length 64 and 65 over F_8. *Information Theory, IEEE Transactions on*, 49(5) :1351–1353, 2003.

[11] R. T. Chien. Cyclic decoding procedures for Bose-Chaudhuri-Hocquenghem codes. *IEEE Transactions on Information Theory*, 10(4) :357–363, 1964.

[12] David Cox, John Littel, and Donal O'Shea. *Ideals, Varieties and Algorithms.* Springer, 1992.

[13] G. L. Feng and T. R. N. Rao. Decoding algebraic-geometric codes up to the designed minimum distance. *Information Theory, IEEE Transactions on*, 39(1) :37–45, 1993.

[14] Gui-Liang Feng and Kenneth K. Tzeng. A generalization of the Berlekamp-Massey algorithm for multisequence shift-register synthesis with applications to decoding cyclic codes. *IEEE Transactions on Information Theory*, 37(5) :1274–1287, 1991.

[15] G. Forney. On decoding BCH codes. *Information Theory, IEEE Transactions on*, 11(4) :549–557, 1965.

[16] Philippe Gaborit and Gilles Zemor. Asymptotic improvement of the Gilbert-Varshamov bound for binary linear codes. In *Information Theory, 2006 IEEE International Symposium on*, pages 287–291, 2006.

[17] Peter Gemmell and Madhu Sudan. Highly resilient correctors for polynomials. *Information Processing Letters*, 43(4) :169–174, 1992.

[18] V. D. Goppa. Codes that are associated with divisors. *Problemy Pereda ci Informacii*, 13(1) :33–39, 1977.

[19] V. Guruswami and M. Sudan. On representations of algebraic-geometry codes. *Information Theory, IEEE Transactions on*, 47(4) :1610–1613, 2001.

[20] V. Guruswami and A. Vardy. Maximum-likelihood decoding of Reed-Solomon codes is NP-hard. *Information Theory, IEEE Transactions on*, 51(7) :2249–2256, 2005.

[21] Venkatesan Guruswami. *Algorithmic results in list decoding*. Now Publishers Inc, January 2007.

[22] Venkatesan Guruswami. List decoding of binary codes - a brief survey of some recent results. In Yeow M. Chee, Chao Li, San Ling, Huaxiong Wang, and Chaoping Xing, editors, *Coding and Cryptology*, volume 5557 of *Lecture Notes in Computer Science*, pages 97–106, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[23] Venkatesan Guruswami and Anindya C. Patthak. Correlated algebraic-geometric codes : Improved list decoding over bounded alphabets. *Mathematics of computation*, September 2007.

[24] Venkatesan Guruswami and Atri Rudra. Explicit capacity-achieving list-decodable codes. In *STOC '06 : Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 1–10, New York, NY, USA, 2006. ACM.

[25] Venkatesen Guruswami and Madhu Sudan. Extensions to the Johnson bound. `http://theory.lcs.mit.edu/Emadhu/papers/johnson.ps`, 2001.

[26] R. W. Hamming. Error detecting and error correcting codes. Technical Report 2, Bell System, April 1950.

[27] Tom Høholdt and Ruud Pellikaan. On the decoding of algebraic-geometric codes. *IEEE Transactions on Information Theory*, 41(6) :1589–1614, 1995.

[28] T. Høholdt, J. H. van Lint, and R. Pellikaan. *Handbook of Coding Theory*, volume I, chapter Algebraic geometry codes, pages 871–961. Elsevier, 1998.

[29] Jørn Justesen and Tom Høholdt. Bounds on list decoding of MDS codes. *IEEE Transactions on Information Theory*, 47(4) :1604–1609, 2001.

[30] R. Kotter. A fast parallel implementation of a Berlekamp-Massey algorithm for algebraic-geometric codes. *IEEE Transactions on Information Theory*, 44(4) :1353–1368, July 1998.

[31] F. J. Macwilliams and N. J. A. Sloane. *The Theory of Error-Correcting Codes*. North-Holland Mathematical Library. North Holland, January 1983.

[32] Jim Massey. Shift-register synthesis and BCH decoding. *IEEE Transactions on Information Theory*, 15(1) :122–127, 1969.

[33] Hajime Matsui, Shojiro Sakata, Masazumi Kurihara, and Seiichi Mita. Systolic array architecture implementing Berlekamp-Massey-Sakata algorithm for decoding codes on a class of algebraic curves. *IEEE Transactions on Information Theory*, 51(11) :3856–3871, 2005.

[34] Robert J. McEliece. The Guruswami-Sudan decoding algorithm for Reed-Solomon codes. Technical Report 42-153, JPL Interplanetary Network Progress Report, May 2003.

[35] K. Saints and C. Heegard. Algebraic-geometric codes and multidimensional cyclic codes : a unified theory and algorithms for decoding using Grobner bases. *Information Theory, IEEE Transactions on*, 41(6) :1733–1751, Nov 1995.

[36] S. Sakata, J. Justesen, Y. Madelung, H. E. Jensen, and T. Hoholdt. Fast decoding of algebraic-geometric codes up to the designed minimum distance. *Information Theory, IEEE Transactions on*, 41(6) :1672–1677, 1995.

[37] Shojiro Sakata. Finding a minimal set of linear recurring relations capable of generating a given finite two-dimensional array. *J. Symb. Comput.*, 5(3) :321–337, 1988.

[38] Shojiro Sakata. N-dimensional Berlekamp-Massey algorithm for multiple arrays and construction of multivariate polynomials with preassigned zeros. In Teo Mora, editor, *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes 6th International Conference, AAECC-6 Rome*, volume 357 of *Lecture Notes in Computer Science*, pages 356–376, 1988.

[39] Shojiro Sakata. Finding a minimal polynomial vector set of a vector of nD arrays. In H. F. Mattson, T. Mora, and T. R. N. Rao, editors, *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, number 539 in Lecture Notes in Computer Science, pages 414–425. Springer-Verlag, 1991.

[40] Eli B. Sasson, Swastik Kopparty, and Jaikumar Radhakrishnan. Subspace polynomials and list decoding of Reed-Solomon codes. In *FOCS '06 : Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pages 207–216, Washington, DC, USA, 2006. IEEE Computer Society.

[41] J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM*, 27(4) :701–717, October 1980.

[42] Claude E. Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27 :379–423, July 1948.

[43] Mohammad A. Shokrollahi and Hal Wasserman. Decoding algebraic geometric codes. In *Proceedings of the IEEE Information Theory Workshop, 1998*, pages 40–41, 1998.

[44] Henning Stichtenoth. *Algebraic Function Fields and Codes*. Springer, 1993.

[45] Madhu Sudan. Decoding of Reed-Solomon codes beyond the error-correction bound. *Journal of Complexity*, 13(1) :180–193, March 1997.

[46] Michael A. Tsfasman, Serguei G. Vlăduţ, and Th. Zink. Modular curves, Shimura curves, and Goppa codes, better than Varshamov-Gilbert bound. *Math. Nachr.*, 109 :21–28, 1982.

[47] Alexander Vardy. Algorithmic complexity in coding theory and the minimum distance problem. In *STOC '97 : Proceedings of the twenty-ninth annual ACM symposium on Theory of computing, El Paso, United States*, pages 92–109. ACM Press, 1997.

[48] Joachim von zur Gathen and Jürgen Gerhard. *Modern Computer Algebra*. Cambridge University Press, 1999.

[49] B. E. Wahlen and J. Jimenez. Performance comparison of Hermitian and Reed-Solomon codes. In *MILCOM 97 Proceedings*, volume 1, pages 15–19 vol.1, 1997.

[50] Loyd R. Welch and Elwyn R. Berlekamp. Error correction for algebraic block codes. US Patent 4 633 470, December 1986.

[51] Y. Wu. New list decoding algorithms for Reed-Solomon and BCH codes. *Information Theory, IEEE Transactions on*, 54(8) :3611–3630, 2008.

[52] Richard Zippel. Probabilistic algorithms for sparse polynomials. In Edward W. Ng, editor, *Symbolic and Algebraic Computation : EUROSAM '79*, volume 72 of *Lecture Notes in Computer Science*, pages 216–226. Springer, 1979.