Chapter 7 Reed-Solomon Codes

- 7.1 Finite Field Algebra
- 7.2 Reed-Solomon Codes
- 7.3 Syndrome Based Decoding
- 7.4 Interpolation Based Decoding

§ 7.1 Finite Field Algebra

- Nonbinary codes: message and codeword symbols are represented in a finite field of size q, and $q>2$.
- Advantage of presenting a code in a nonbinary image.

A binary codeword sequence in {0,1}

b_0 $\boxed{b_1}$ $\boxed{b_2}$ $\boxed{b_3}$ $\boxed{b_4}$ $\boxed{b_5}$ b_6 b_7 b_8 b_9 b_{10} b_{11} $\boxed{b_{12}}$ $\boxed{b_{13}}$ $\boxed{b_{14}}$ b_{15} b_{16} b_{17}

b_{18} b_{19} b_{20}

A nonbinary codeword sequence in {0, 1, 2, 3, 4, 5, 6, 7}

$\boxed{c_0}$ $\boxed{c_1}$ c_2 c_3 $\boxed{c_4}$ c_5 c_6

$\boxed{}$: where the channel error occurs

8 bit errors are treated as 3 symbol errors in a nonbinary image

§ 7.1 Finite Field Algebra

– Finite field (Galois field) $\mathbf{F}_q$: a set of q elements that perform " + " " - " " × " " / " without leaving the set.
– Let p denote a prime, e.g., 2, 3, 5, 7, 11, ⋯ , it is required $q = p$ *or* $q = p^\theta$ (θ is a positive integer greater than 1). If $q = p^\theta$, $\mathbf{F}_q$ is an extension field of $\mathbf{F}_p$.
– ***Example 7.1*****:** " + " and " × " in $\mathbf{F}_q$.

$$\mathbf{F}_2 = \{ 0, 1 \}$$

+	0	1
0	0	1
1	1	0

×	0	1
0	0	0
1	0	1

all in modulo-2

$$\mathbf{F}_5 = \{ 0, 1, 2, 3, 4 \}$$

+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

×	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

all in modulo-5

§ 7.1 Finite Field Algebra

– " - " and " / " can be performed as " + " and " × " with additive inverse and multiplicative inverse, respectively.

Additive inverse of *a* a': $a' + a = 0$ and $a' = -a$

Multiplicative inverse of *a* a': $a' \cdot a = 1$ and $a' = 1 / a$

– " - " operation:

Let $h, a \in \mathbf{F}_q$.

$h - a = h + (-a) = h + a'$.

E.g., in $\mathbf{F}_5$, $1 - 3 = 1 + (-3) = 1 + 2 = 3$;

– " / " operation:

Let $h, a \in \mathbf{F}_q$.

$h / a = h \times a'$.

E.g., in $\mathbf{F}_5$, $2 / 3 = 2 \times (1 / 3) = 2 \times 2 = 4$.

§ 7.1 Finite Field Algebra

– Nonzero elements of $\mathbf{F}_q$ can be represented using a primitive element σ such that $\mathbf{F}_q=\{ 0, 1, \sigma, \sigma^2, \cdots , \sigma^{q-2} \}$.

– Primitive element σ of $\mathbf{F}_q$: $\sigma \in \mathbf{F}_q$ and unity can be produced by at least

$$\underbrace{\sigma \cdot \sigma \cdots \sigma}_{q-1} = 1, \text{ or } \sigma^{q-1} = 1.$$

all in modulo-*q*

E.g., in $\mathbf{F}_5$, $2^4 = 1$ and $3^4 = 1$. Here, 2 and 3 are the primitive elements of $\mathbf{F}_5$.

– ***Example 7.2*****:** In $\mathbf{F}_5$,

If 2 is chosen as the primitive element, then

$$\mathbf{F}_5 = \{ 0, 1, 2, 3, 4 \} = \{0, 2^0, 2^1, 2^3, 2^2 \}= \{0, 1, 2^1, 2^3, 2^2 \}$$

If 3 is chosen as the primitive element, then

$$\mathbf{F}_5 = \{ 0, 1, 2, 3, 4 \} = \{0, 3^0, 3^3, 3^1, 3^2 \} = \{0, 1, 3^3, 3^1, 3^2 \}$$

§ 7.1 Finite Field Algebra

– If $\mathbf{F}_q$ is an extension field of $\mathbf{F}_p$ such as $q = p^\theta$, elements of $\mathbf{F}_q$ can also be represented by θ-dimensional vectors in $\mathbf{F}_p$.

– Primitive polynomial $p(x)$ of $\mathbf{F}_q$ ($q = p^\theta$): an irreducible polynomial of degree θ that divides $x^{p^\theta - 1} - 1$ but not other polynomials x^Φ - 1 with $\Phi < p^\theta$ - 1.
E.g., in $\mathbf{F}_8$, the primitive polynomial $p(x) = x^3 + x + 1$ divides x^7-1, but not x^6-1, x^5-1, x^4-1, x^3-1.

– If variable δ is a root of $p(x)$ such that $p(\delta) = 0$, elements of $\mathbf{F}_q$ can be represented in the form of

$$w_{\theta\text{-}1}\delta^{\theta\text{-}1} + w_{\theta\text{-}2}\delta^{\theta\text{-}2} + \ldots + w_1\delta^1 + w_0\delta^0$$

where $w_0, w_1, \ldots, w_{\theta\text{-}2}, w_{\theta\text{-}1} \in \mathbf{F}_p$, or alteratively in

$$(w_{\theta\text{-}1}, w_{\theta\text{-}2}, \cdots, w_1, w_0)$$

§ 7.1 Finite Field Algebra

– ***Example 7.3*:** If $p(x) = x^3 + x + 1$ is the primitive polynomial of $\mathbf{F}_8$, and symbol δ satisfies $\delta^3 + \delta + 1 = 0$, then

$\mathbf{F}_8$	$w_2\delta^2 + w_1\delta^1 + w_0\delta^0$	w_2 w_1 w_0
0	0	0 0 0
1	1	0 0 1
σ	δ	0 1 0
σ^2	δ^2	1 0 0
σ^3	$\delta + 1$	0 1 1
σ^4	$\delta^2 + \delta$	1 1 0
σ^5	$\delta^2 + \delta + 1$	1 1 1
σ^6	$\delta^2 + 1$	1 0 1

§ 7.1 Finite Field Algebra

– Representing $\mathbf{F}_q = \{ 0, 1, \sigma, \cdots, \sigma^{q-2} \}$, " × " " / " " + " " - " operations become

" × ": $\sigma^i \times \sigma^j = \sigma^{(i+j) \% (q-1)}$

E.g., in $\mathbf{F}_8$, $\sigma^4 \times \sigma^5 = \sigma^{(4+5) \% 7} = \sigma^2$

" / ": $\sigma^i / \sigma^j = \sigma^{(i-j) \% (q-1)}$

E.g., in $\mathbf{F}_8$, $\sigma^4 / \sigma^5 = \sigma^{(4-5) \% 7} = \sigma^6$

" + ": if $\sigma^i = w_{\theta-1}\delta^{\theta-1} + w_{\theta-2}\delta^{\theta-2} + \cdots + w_0\delta^0$

(&" - ") $\sigma^j = w'_{\theta-1}\delta^{\theta-1} + w'_{\theta-2}\delta^{\theta-2} + \cdots + w'_0\delta^0$

$\sigma^i + \sigma^j = (w_{\theta-1} + w'_{\theta-1})\delta^{\theta-1} + (w_{\theta-2} + w'_{\theta-2})\delta^{\theta-2} + \cdots + (w_0 + w'_0)\delta^0$

E.g., in $\mathbf{F}_8$, $\sigma^4 + \sigma^5 = \delta^2 + \delta + \delta^2 + \delta + 1 = 1$

§ 7.2 Reed-Solomon Codes

– An RS code[1] defined over $\mathbf{F}_q$ is characterized by its codeword length $n = q - 1$, dimension $k < n$ and the minimum Hamming distance d. It is often denoted as an (n, k) (or (n, k, d)) RS code.

– It is a maximum distance separable (MDS) code such that

$$d = n - k + 1$$

– It is a linear block code and also cyclic.

– The widely used RS codes include the (255, 239) and the (255, 223) codes both of which are defined in $\mathbf{F}_{256}$.

[1] I. Reed and G. Solomon, "Polynomial codes over certain finite fields," *J. Soc. Indust. Appl. Math*, vol. 8, pp. 300-304, 1960.

§ 7.2 Reed-Solomon Codes

– Notations

$\mathbf{F}_q[x]$, a univariate polynomial ring over $\mathbf{F}_q$, i.e., $f(x)=\sum_{i\in\mathrm{N}} f_i x^i$ and $f_i \in \mathbf{F}_q$.

$\mathbf{F}_q[x, y]$, a bivariate polynomial ring over $\mathbf{F}_q$, i.e., $f(x,y)=\sum_{i,j\in\mathrm{N}} f_{ij} x^i y^j$ and $f_{ij} \in \mathbf{F}_q$.

$\mathbf{F}_q^{\bullet}$, • - dimensional vector over $\mathbf{F}_q$.

– Encoding of an (n, k) RS code.

Message vector $\bar{u}=(u_0,u_1,u_2,\cdots,u_{k-1})\in\mathbf{F}_q^k$

Message polynomial

$$u(x)=u_0+u_1x+u_2x^2+\cdots+u_{k-1}x^{k-1}\in\mathbf{F}_q[x]$$

Codeword

$$\bar{c}=(u(1),u(\sigma),u(\sigma^2),\cdots,u(\sigma^{n-1}))\in\mathbf{F}_q^n$$

$1,\sigma,\sigma^2,\cdots,\sigma^{n-1}$ are the q - 1 nonzero elements of $\mathbf{F}_q$. They are often called code locators. Note that the above evaluation order can be arbitrary.

§ 7.2 Reed-Solomon Codes

– Encoding of an (n, k) RS code in a linear block code fashion

$$\bar{c} = \bar{u} \cdot \mathbf{G}$$

$$= (u_0, u_1, \cdots, u_{k-1}) \begin{bmatrix} (\sigma^0)^0 & (\sigma^1)^0 & \cdots & (\sigma^{n-1})^0 \\ (\sigma^0)^1 & (\sigma^1)^1 & \cdots & (\sigma^{n-1})^1 \\ \vdots & \vdots & \ddots & \vdots \\ (\sigma^0)^{k-1} & (\sigma^1)^{k-1} & \cdots & (\sigma^{n-1})^{k-1} \end{bmatrix}$$

– ***Example 7.4*****:** For a (7, 3) RS code that is defined in $\mathbf{F}_8$, if the message is $\bar{u} = (u_0, u_1, u_2) = (0, \sigma, \sigma^6)$, the message polynomial will be $u(x) = \sigma x + \sigma^6 x^2$, and the codeword can be generated by

- $\bar{c} = (u(1), u(\sigma), u(\sigma^2), u(\sigma^3), u(\sigma^4), u(\sigma^5), u(\sigma^6)) = (\sigma^5, \sigma^4, 0, 1, \sigma^4, 1, \sigma^5)$
- $\bar{c} = \bar{u} \cdot \mathbf{G} = \left(0, \sigma, \sigma^6\right) \cdot \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \sigma^1 & \sigma^2 & \sigma^3 & \sigma^4 & \sigma^5 & \sigma^6 \\ 1 & \sigma^2 & \sigma^4 & \sigma^6 & \sigma^1 & \sigma^3 & \sigma^5 \end{bmatrix} = (\sigma^5, \sigma^4, 0, 1, \sigma^4, 1, \sigma^5)$

§ 7.2 Reed-Solomon Codes

- MDS property of RS codes $d = n - k + 1$
  - Singleton bound for an (n, k) linear block code, $d \le n - k + 1$
  - $u(x)$ has at most k - 1 roots. Hence, $\bar{c}$ has at most k - 1 zeros and

$$d_{\text{Ham}} = (\bar{c}, \bar{0}) \ge n - k + 1$$

- Parity-check matrix of an (n, k) RS code

$$\mathbf{H} = \begin{bmatrix} (\sigma^0)^1 & (\sigma^1)^1 & \cdots & (\sigma^{n-1})^1 \\ (\sigma^0)^2 & (\sigma^1)^2 & \cdots & (\sigma^{n-1})^2 \\ \vdots & \vdots & \ddots & \vdots \\ (\sigma^0)^{n-k} & (\sigma^1)^{n-k} & \cdots & (\sigma^{n-1})^{n-k} \end{bmatrix}$$

$\bar{c} \cdot \mathbf{H}^T = \bar{u} \cdot \mathbf{G} \cdot \mathbf{H}^T = \bar{0}$ ← an n - k all zero vector

§ 7.2 Reed-Solomon Codes

– Insight of $\mathbf{G}\cdot\mathbf{H}^T$

$$\begin{bmatrix} (\sigma^0)^0 & (\sigma^1)^0 & \cdots & (\sigma^{n-1})^0 \\ (\sigma^0)^1 & (\sigma^1)^1 & \cdots & (\sigma^{n-1})^1 \\ \vdots & \vdots & \ddots & \vdots \\ (\sigma^0)^{k-1} & (\sigma^1)^{k-1} & \cdots & (\sigma^{n-1})^{k-1} \end{bmatrix} \cdot \begin{bmatrix} (\sigma^0)^1 & (\sigma^0)^2 & \cdots & (\sigma^0)^{n-k} \\ (\sigma^1)^1 & (\sigma^1)^2 & \cdots & (\sigma^1)^{n-k} \\ \vdots & \vdots & \ddots & \vdots \\ (\sigma^{n-1})^1 & (\sigma^{n-1})^2 & \cdots & (\sigma^{n-1})^{n-k} \end{bmatrix}$$

– Let $i = 0, 1, \cdots, k-1$, $j = 0, 1, \cdots, n - 1$, $v = 1, 2, \cdots, n - k$.

Entries of **G** can be denoted as $[\,\mathbf{G}\,]_{i,j} = (\,\sigma^j\,)^i$

Entries of $\mathbf{H}^T$ can be denoted as $[\,\mathbf{H}^T\,]_{j,v-1} = (\,\sigma^j\,)^v$

Entries of $\mathbf{G}\cdot\mathbf{H}^T$ is

$$[\mathbf{G}\cdot\mathbf{H}^T]_{i,v-1} = \sum_{j=0}^{n-1}(\sigma^j)^i \cdot (\sigma^j)^v$$
$$= \sum_{j=0}^{n-1}(\sigma^j)^{i+v} = 0$$

Remark 1: $v = 0$ is illegitimate since $\sum_{j=0}^{n-1}(\sigma^j)^0 \neq 0$

§ 7.2 Reed-Solomon Codes

– Perceiving $\mathbf{H}^T$ as in

$$\begin{bmatrix} (\sigma^1)^0 & (\sigma^2)^0 & \cdots & (\sigma^{n-k})^0 \\ (\sigma^1)^1 & (\sigma^2)^1 & \cdots & (\sigma^{n-k})^1 \\ \vdots & \vdots & \ddots & \vdots \\ (\sigma^1)^{n-1} & (\sigma^2)^{n-1} & \cdots & (\sigma^{n-k})^{n-1} \end{bmatrix}$$

– Perceiving codeword $\bar{c} = (c_0, c_1, \cdots, c_{n-1})$ as in

$$c(x) = c_0 + c_1 x + \cdots + c_{n-1} x^{n-1}$$

– $\bar{c} \cdot \mathbf{H}^T = \bar{0}$ implies

$$c(\sigma^1) = c(\sigma^2) = \cdots = c(\sigma^{n-k}) = 0$$

$\sigma^1, \sigma^2, \cdots, \sigma^{n-k}$ are roots of RS codeword polynomial $c(x)$.

§ 7.2 Reed-Solomon Codes

– An alternative encoding
  – Message polynomial $u(x) = u_0 + u_1 x + \cdots + u_{k-1} x^{k-1}$
  – Codeword polynomial $c(x) = c_0 + c_1 x + \cdots + c_{n-1} x^{n-1}$
  – $c(x) = u(x) \cdot g(x)$ and deg(g(x)) = n - k
  – Since $\sigma^1, \sigma^2, \cdots, \sigma^{n-k}$ are roots of $c(x)$

$$g(x) = (x - \sigma^1)(x - \sigma^2)\cdots(x - \sigma^{n-k})$$

↑ The generator polynomial of an (n, k) RS code

  – Systematic encoding

$$c(x) = x^{n-k} u(x) + (x^{n-k} u(x)) \bmod g(x)$$

– ***Example 7.5*****:** For a (7, 3) RS code, its generator polynomial is

$$g(x) = (x - \sigma^1)(x - \sigma^2)(x - \sigma^3)(x - \sigma^4) = x^4 + \sigma^3 x^3 + x^2 + \sigma x + \sigma^3$$

Given message vector $\bar{u} = (u_0, u_1, u_2) = (\sigma^4, 1, \sigma^5)$,

the codeword can be generated by $c(x) = u(x) \cdot g(x) = (1, \sigma^2, \sigma^4, \sigma^6, \sigma, \sigma^3, \sigma^5)$

For systematic encoding, $\left(x^{n-k} u(x)\right) \bmod g(x) = \left(x^4 \cdot u(x)\right) \bmod g(x) = x^3 + \sigma^4 x + \sigma^5$,

and the codeword is $\bar{c} = (\sigma^5, \sigma^4, 0, 1, \sigma^4, 1, \sigma^5)$

§ 7.3 Syndrome Based Decoding

– The channel: $r(x) = c(x) + e(x)$

$c(x) = c_0 + c_1 x + \cdots + c_{n-1} x^{n-1}$ – codeword polynomial

$e(x) = e_0 + e_1 x + \cdots + e_{n-1} x^{n-1}$ – error polynomial

$r(x) = r_0 + r_1 x + \cdots + r_{n-1} x^{n-1}$ – received word polynomial

– Let $n - k = 2t$, $\sigma^1, \sigma^2, \cdots, \sigma^{2t}$ are roots of $c(x)$

– $2t$ syndromes can be determined as

$$S_1 = r(\sigma^1), S_2 = r(\sigma^2), \cdots, S_{2t} = r(\sigma^{2t})$$

If $S_1 = S_2 = \cdots = S_{2t} = 0$, $r(x)$ is a valid codeword. Otherwise, $e(x) \neq 0$, error-correction is needed.

§ 7.3 Syndrome Based Decoding

- If $e(x) \neq 0$, we assume there are ω errors with $e_{j_1} \neq 0, e_{j_2} \neq 0, \cdots, e_{j_\omega} \neq 0$.
- Let v = 1, 2, ⋯ ,2t

$$S_v = \sum_{j=0}^{n-1} c_j \sigma^{jv} + \sum_{j=0}^{n-1} e_j \sigma^{jv} = \sum_{j=0}^{n-1} e_j \sigma^{jv} = \sum_{\tau=1}^{\omega} e_{j_\tau} (\sigma^{j_\tau})^v$$

- For simplicity, let $X_\tau = \sigma^{j_\tau}$, we can list the 2t syndromes by

$$\begin{aligned} S_1 &= e_{j_1} X_1^1 + e_{j_2} X_2^1 + \cdots + e_{j_\omega} X_\omega^1 \\ S_2 &= e_{j_1} X_1^2 + e_{j_2} X_2^2 + \cdots + e_{j_\omega} X_\omega^2 \\ &\vdots \\ S_{2t} &= e_{j_1} X_1^{2t} + e_{j_2} X_2^{2t} + \cdots + e_{j_\omega} X_\omega^{2t} \end{aligned}$$

- In the above non-linear equation group, there are 2ω unknowns $X_1, X_2, \cdots, X_\omega$, $e_{j_1}, e_{j_2}, \cdots, e_{j_\omega}$. It will be solvable if $2\omega \leq 2t$. The number of correctable errors is $\omega \leq \frac{n-k}{2}$.
- Since $X_{j_\tau}, e_{j_\tau} \in \mathbf{F}_q \setminus \{0\}$, an exhaustive search solution will have a complexity of $O(n^{2\omega})$.

§ 7.3 Syndrome Based Decoding

– In order to decode an RS code with a polynomial-time complexity, the decoding is decomposed into determining the **error locations** and **error magnitudes**, i.e., $X_1, X_2, \cdots, X_\omega$ and $e_{j_1}, e_{j_2}, \cdots, e_{j_\omega}$, respectively.

– Error locator polynomial

$$\Lambda(x) = \prod_{\tau=1}^{\omega} (1 - X_\tau x)$$

$$= \Lambda_\omega x^\omega + \Lambda_{\omega-1} x^{\omega-1} + \cdots + \Lambda_1 x + \Lambda_0 \quad (\Lambda_0 = 1)$$

$X_1^{-1} = \sigma^{-j_1}, X_2^{-1} = \sigma^{-j_2}, \cdots, X_\omega^{-1} = \sigma^{-j_\omega}$ are roots of the polynomial such that $\Lambda(X_1^{-1}) = \Lambda(X_2^{-1}) = \cdots = \Lambda(X_\omega^{-1}) = 0$.

– Determine $\Lambda(x)$ by finding out $\Lambda_\omega, \Lambda_{\omega-1}, \cdots,$ and Λ_1, and its roots tell the error locations.

§ 7.3 Syndrome Based Decoding

– How to determine $\Lambda_{\omega}, \Lambda_{\omega-1}, \cdots,$ and Λ_1?

Since $\Lambda(X_{\tau}^{-1}) = \Lambda_{\omega} X_{\tau}^{-\omega} + \Lambda_{\omega-1} X_{\tau}^{1-\omega} + \cdots + \Lambda_1 X_{\tau}^{-1} + \Lambda_0 = 0$

$$\sum_{\tau=1}^{\omega} e_{j_\tau} X_{\tau}^{v} \Lambda(X_{\tau}^{-1}) = 0, \text{ for } v = 1, 2, \cdots, 2t$$

$$\Downarrow$$

$$= e_{j_1} \Lambda_{\omega} X_1^{v-\omega} + e_{j_1} \Lambda_{\omega-1} X_1^{v-\omega+1} + \cdots + e_{j_1} \Lambda_1 X_1^{v-1} + e_{j_1} \Lambda_0 X_1^{v}$$

$$+ e_{j_2} \Lambda_{\omega} X_2^{v-\omega} + e_{j_2} \Lambda_{\omega-1} X_2^{v-\omega+1} + \cdots + e_{j_2} \Lambda_1 X_2^{v-1} + e_{j_2} \Lambda_0 X_2^{v}$$

$$\vdots$$

$$+ e_{j_\omega} \Lambda_{\omega} X_{\omega}^{v-\omega} + e_{j_\omega} \Lambda_{\omega-1} X_{\omega}^{v-\omega+1} + \cdots + e_{j_\omega} \Lambda_1 X_{\omega}^{v-1} + e_{j_\omega} \Lambda_0 X_{\omega}^{v}$$

$$= \Lambda_{\omega} S_{v-\omega} + \Lambda_{\omega-1} S_{v-\omega+1} + \cdots + \Lambda_1 S_{v-1} + \Lambda_0 S_v$$

$$\boxed{\Lambda_{\omega} S_{v-\omega} + \Lambda_{\omega-1} S_{v-\omega+1} + \cdots + \Lambda_1 S_{v-1} + \Lambda_0 S_v = 0}$$

– Error locator polynomial can be determined using the syndromes.

§ 7.3 Syndrome Based Decoding

– List all $\Lambda_\omega S_{v-\omega} + \Lambda_{\omega-1} S_{v-\omega+1} + \cdots + \Lambda_1 S_{v-1} + \Lambda_0 S_v = 0$

$v = 1$: $\Lambda_1 S_0 + \Lambda_0 S_1 = \cdot\cdot$

$v = 2$: $\Lambda_2 S_0 + \Lambda_1 S_1 + \Lambda_0 S_2 = \cdot\cdot$

$v = 3$: $\Lambda_3 S_0 + \Lambda_2 S_1 + \Lambda_1 S_2 + \Lambda_0 S_3 = \cdot\cdot$

$\vdots$

$v = \omega$: $\Lambda_\omega S_0 + \Lambda_{\omega-1} S_1 + \cdots + \Lambda_1 S_{\omega-1} + \Lambda_0 S_\omega = \cdot\cdot$

$v = \omega + 1$: $\Lambda_\omega S_1 + \Lambda_{\omega-1} S_2 + \cdots + \Lambda_1 S_\omega + \Lambda_0 S_{\omega+1} = 0$

$v = \omega + 2$: $\Lambda_\omega S_2 + \Lambda_{\omega-1} S_3 + \cdots + \Lambda_1 S_{\omega+1} + \Lambda_0 S_{\omega+2} = 0$

$\vdots$

$v = 2t$: $\Lambda_\omega S_{2t-\omega} + \Lambda_{\omega-1} S_{2t-\omega+1} + \cdots + \Lambda_1 S_{2t-1} + \Lambda_0 S_{2t} = 0$

$$S_v = -\sum_{\tau=1}^{\omega} \Lambda_\tau S_{v-\tau}$$

$$\begin{bmatrix} S_1 & S_2 & \cdots & S_\omega \\ S_2 & S_3 & \cdots & S_{\omega+1} \\ \vdots & \vdots & \ddots & \vdots \\ S_{2t-\omega} & S_{2t-\omega+1} & \cdots & S_{2t-1} \end{bmatrix} \cdot \begin{bmatrix} \Lambda_\omega \\ \Lambda_{\omega-1} \\ \vdots \\ \Lambda_1 \end{bmatrix} = - \begin{bmatrix} S_{\omega+1} \\ S_{\omega+2} \\ \vdots \\ S_{2t} \end{bmatrix}$$

Remark 2:
S_0 is not one of the n - k syndromes.

§ 7.3 Syndrome Based Decoding

– Solving the linear system in finding $\Lambda(x)$ has a complexity of $O(\omega^3)$. It can be facilitated by the Berlekamp-Massey algorithm[2] whose complexity is $O(\omega^2)$.

– The Berlekamp-Massey algorithm can be implemented using the Linear Feedback Shift Register. Its pseudo program is the follows.

The Berlekamp-Massey Algorithm

Input: Syndromes $S_1, S_2, \ldots, S_{2t}$;
Output: $\Lambda(x)$;
Initialization: $r = 0, \ell = 0, z = -1, \Lambda(x) = 1, T(x) = x$;

1: Determine $\Delta = \sum_{i=0}^{\ell} \Lambda_i S_{r-i+1}$;
2: **If** $\Delta = 0$
3: $T(x) = xT(x)$;
4: $r = r + 1$;
5: **If** $r < 2t$
6: Go to **1**;
7: **Else**
8: Terminate the algorithm;
9: **Else**
10: Update $\Lambda^*(x) = \Lambda(x) - \Delta T(x)$;
11: **If** $\ell \geq r - z$
12: $\Lambda(x) = \Lambda^*(x)$;
13: **Else**
14: $\ell^* = r - z$; $z = r - \ell$; $T(x) = \Lambda(x) / \Delta$; $\ell = \ell^*$; $\Lambda(x) = \Lambda^*(x)$;
15: $T(x) = xT(x)$;
16: $r = r + 1$;
17: **If** $r < 2t$
18: Go to **1**;
19: **Else**
20: Terminate the algorithm;

[2] J. L. Massey, "Shift register synthesis and BCH decoding," *IEEE Trans. Inf. Theory*, vol. 15(1), pp. 122-127, 1969.

§ 7.3 Syndrome Based Decoding

– ***Example 7.6***: Given the (7, 3) RS codeword generated in ***Example 7.5***, after the channel, the received word is

$$\bar{r} = (\sigma^5, \sigma^4, \boxed{\sigma^3}, \sigma^0, \sigma^4, \boxed{\sigma^2}, \sigma^5).$$

With the received word, we can calculate syndromes as

$$S_1 = r(\sigma) = \sigma^0, S_2 = r(\sigma^2) = \sigma^6, S_3 = r(\sigma^3) = \sigma^6, S_4 = r(\sigma^4) = \sigma^0.$$

Running the above Berlekamp-Massey algorithm,we obtain

r	ℓ	z	$\Lambda(x)$	$T(x)$	Δ
0	0	–1	1	x	σ^0
1	1	0	$1-x$	x	σ^2
2	1	0	$1-\sigma^6 x$	x^2	σ
3	2	1	$1-\sigma^6 x-\sigma x^2$	$\sigma^6 x-\sigma^5 x^2$	σ^5
4			$1-\sigma^3 x-x^2$	$\sigma^6 x^2-\sigma^5 x^3$	

Therefore, the error locator polynomial is $\Lambda(x) = 1-\sigma^3 x - x^2$. In $\mathbf{F}_8$, σ^5 and σ^2 are its roots. Therefore, r_2 and r_5 are corrupted.

§ 7.3 Syndrome Based Decoding

– Determine the error magnitudes $e_{j_1}, e_{j_2}, \cdots, e_{j_\omega}$, so that the erroneous symbols can be corrected by

$$c_{j_1} = r_{j_1} - e_{j_1},\ c_{j_2} = r_{j_2} - e_{j_2},\ \cdots,\ c_{j_\omega} = r_{j_\omega} - e_{j_\omega}$$

– The syndromes $S_v = \sum_{\tau=1}^{\omega} e_{j_\tau} X_\tau^v$, $v = 1, 2, \cdots, 2t$. Knowing $X_1 = \sigma^{j_1}, X_2 = \sigma^{j_2}, \cdots, X_\omega = \sigma^{j_\omega}$ from the error location polynomial $\Lambda(x)$, the above syndrome definition implies

$$\begin{bmatrix} X_1^1 & X_2^1 & \cdots & X_\omega^1 \\ X_1^2 & X_2^2 & \cdots & X_\omega^2 \\ \vdots & \vdots & \ddots & \vdots \\ X_1^{2t} & X_2^{2t} & \cdots & X_\omega^{2t} \end{bmatrix} \begin{bmatrix} e_{j_1} \\ e_{j_2} \\ \vdots \\ e_{j_\omega} \end{bmatrix} = \begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_{2t} \end{bmatrix}$$

– Error magnitudes can be determined from the above set of linear equations.

§ 7.3 Syndrome Based Decoding

– The linear equation set can be efficiently solved using Forney's algorithm.

– Syndrome polynomial

$$S(x) = S_1 + S_2 x + \cdots + S_{2t} x^{2t-1} = \sum_{v=1}^{2t} S_v x^{v-1}$$

– Error evaluation polynomial (The key equation)

$$\Omega(x) = S(x) \cdot \Lambda(x) \bmod x^{2t}$$

– Formal derivative of $\Lambda(x) = \Lambda_\omega x^\omega + \Lambda_{\omega-1} x^{\omega-1} + \cdots + \Lambda_1 x + \Lambda_0$

$$\Lambda'(x) = \underline{\omega \Lambda_\omega} x^{\omega-1} + \underline{(\omega-1)\Lambda_{\omega-1}} x^{\omega-2} + \cdots + \Lambda_1$$

$$\Downarrow \qquad\qquad \Downarrow$$

$$\underbrace{\Lambda_\omega + \Lambda_\omega + \cdots + \Lambda_\omega}_{\omega} \qquad \underbrace{\Lambda_{\omega-1} + \Lambda_{\omega-1} + \cdots + \Lambda_{\omega-1}}_{\omega-1}$$

– Error magnitude e_{j_τ} can be determined by $\boxed{e_{j_\tau} = -\dfrac{\Omega\left(X_\tau^{-1}\right)}{\Lambda'\left(X_\tau^{-1}\right)}}$.

§ 7.3 Syndrome Based Decoding

– ***Example 7.7*:** Continue from ***Example 7.6***,

The syndrome polynomial is $S(x) = S_1 + S_2 x + S_3 x^2 + S_4 x^3 = \sigma^0 + \sigma^6 x + \sigma^6 x^2 + \sigma^0 x^3$.

The error locator polynomial is $\Lambda(x) = 1 - \sigma^3 x - x^2$.

The error evaluation polynomial is $\Omega(x) = S(x) \cdot \Lambda(x) \bmod x^4 = \sigma^4 x + \sigma^0$.

Formal derivative of $\Lambda(x)$ is $\Lambda'(x) = \sigma^3$.

Error magnitudes are

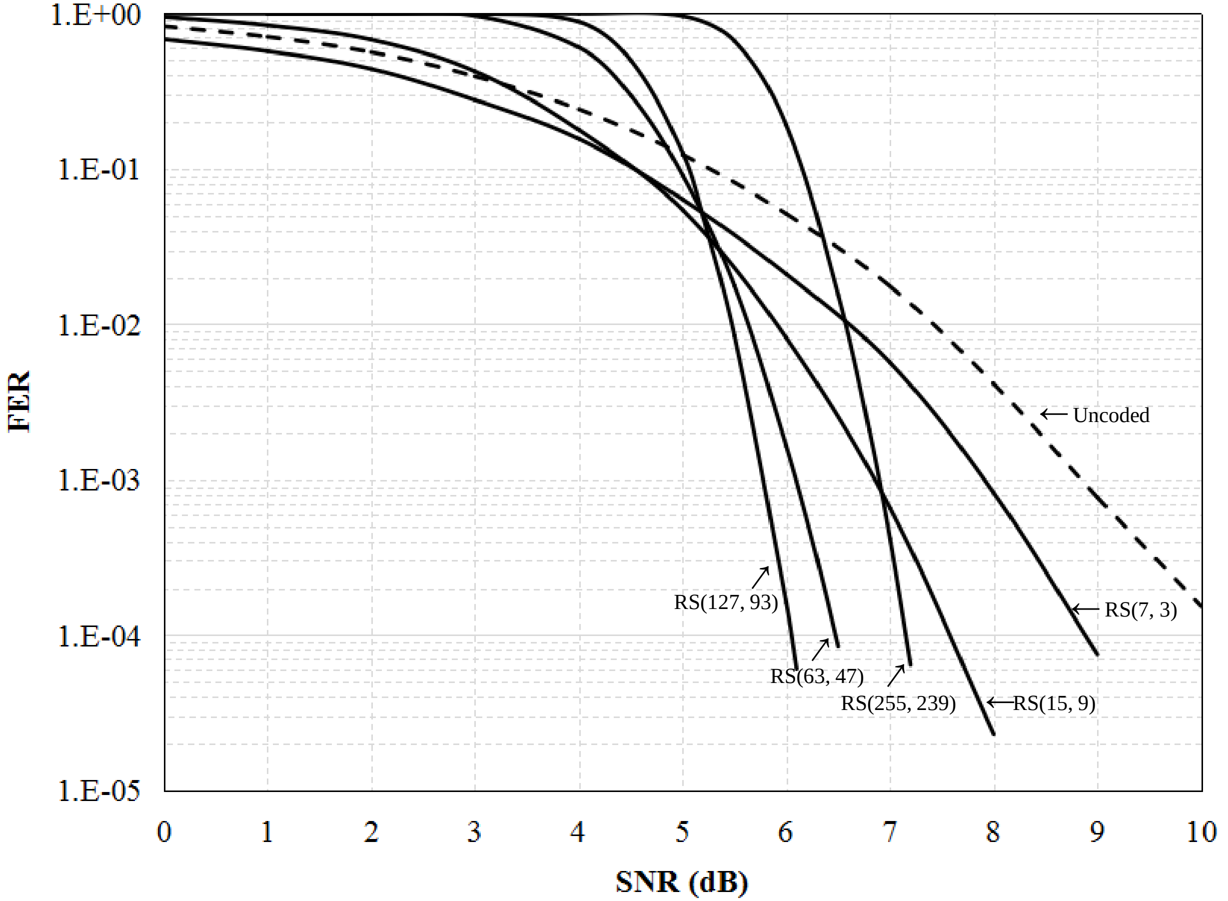
$$e_2 = -\frac{\Omega\left(\sigma^{-2}\right)}{\Lambda'\left(\sigma^{-2}\right)} = \sigma^3 \quad ,$$

$$e_5 = -\frac{\Omega\left(\sigma^{-5}\right)}{\Lambda'\left(\sigma^{-5}\right)} = \sigma^6 \quad .$$

As a result, $c_2 = r_2 - e_2 = 0$, $c_5 = r_5 - e_5 = \sigma^0$.

§ 7.3 Syndrome Based Decoding

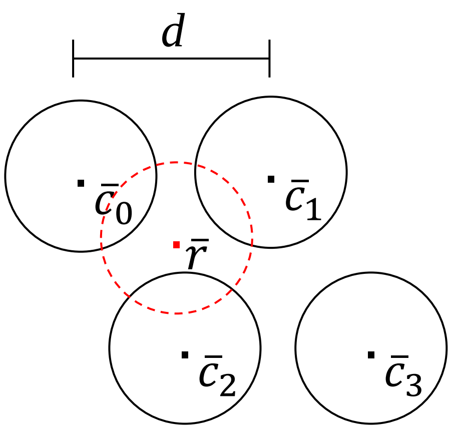
– BM decoding performances over AWGN channel with BPSK.



§ 7.4 Interpolation Based Decoding

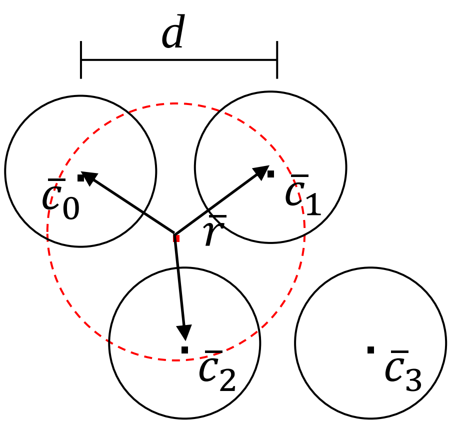
– Error-correction capability

Bounded minimum distance decoding: BM algorithm



$$\tau_{\mathrm{BM}} = \left\lfloor \frac{d-1}{2} \right\rfloor = \left\lfloor \frac{n-k}{2} \right\rfloor$$

List decoding: Guruswami-Sudan (GS) algorithm [3]



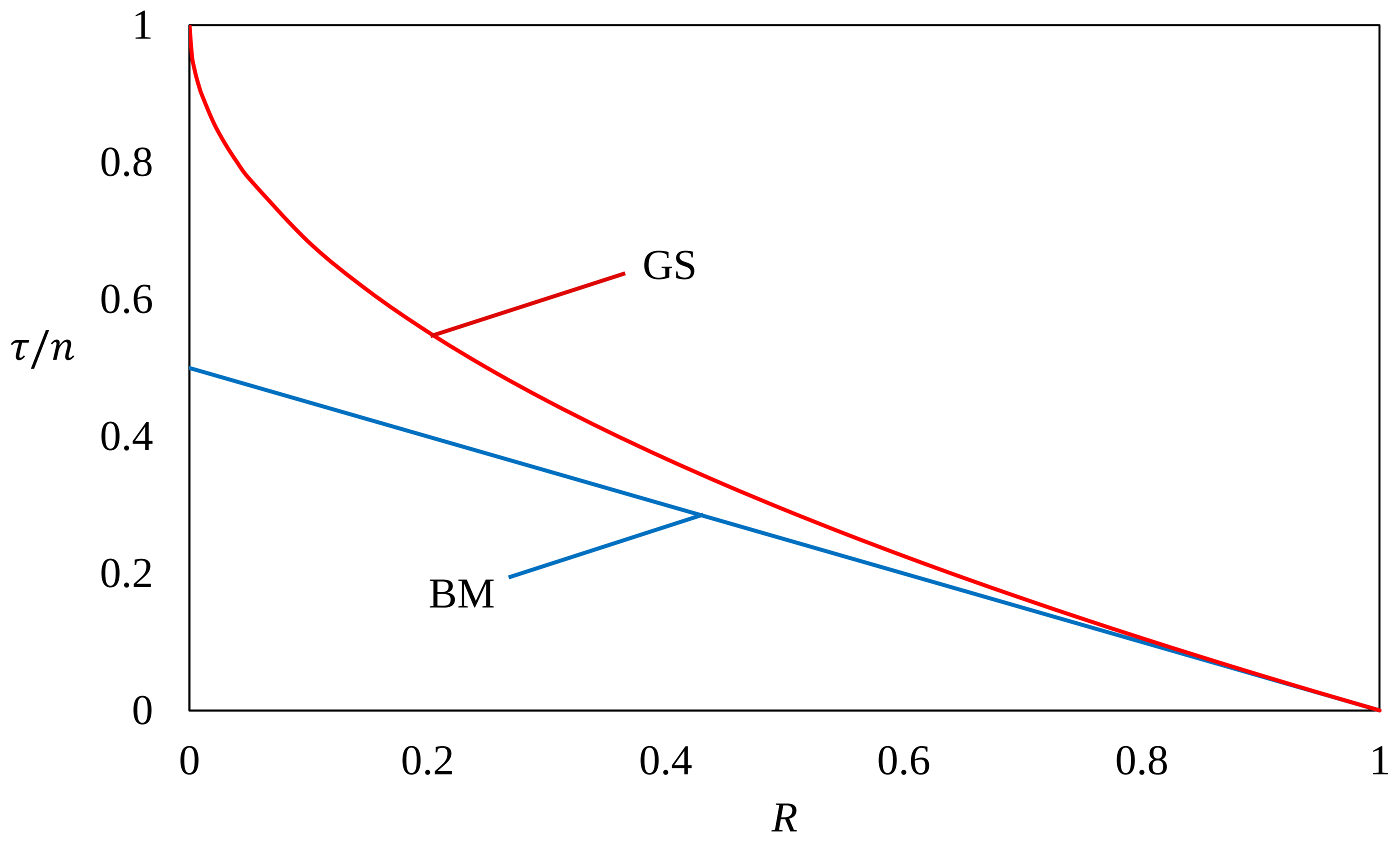
$$\tau_{\mathrm{GS}} = n - \left\lfloor \sqrt{n(k-1)} \right\rfloor - 1 \geq \tau_{\mathrm{BM}}$$

[3] V. Guruswami and M. Sudan, “Improved decoding of Reed-Solomon and algebraic-geometric codes,” *IEEE Trans. Inform. Theory*, vol. 45, no. 6, pp. 1757-1767, Sept. 1999.

§ 7.4 Interpolation Based Decoding

– Fraction of number of correctable errors

$$\frac{\tau_{\mathrm{BM}}}{n} = \frac{1-R}{2} \qquad \frac{\tau_{\mathrm{GS}}}{n} = 1-\sqrt{R}$$



§ 7.4 Interpolation Based Decoding

– GS algorithm

Code locators: $\bar{x} = \{x_0, x_1, \dots \dots, x_{n-1}\}$

Received word: $\bar{r} = \{r_0, r_1, \dots \dots, r_{n-1}\}$

Interpolation points: $(x_0, r_0), (x_1, r_1), \cdots \cdots, (x_{n-1}, r_{n-1})$

– **Interpolation**: Generate the minimum bivariate polynomial $Q(x, y)$ that interpolates the *n* points with a multiplicity of *m*.

– **Factorization**: Find the *y*-roots of $Q(x, y)$ such that a list of polynomials can be obtained as

$$L = \{f(x): Q(x, f(x)) = 0 \text{ and } \deg f(x) < k\}.$$

All the polynomials in *L* have the possibility of being the transmitted message $u(x)$.

– Interpolation dominates the GS decoding complexity.

§ 7.4 Interpolation Based Decoding

– What is "**multiplicity**"?

– Given a polynomial $Q(x,y) = \sum_{a,b} Q_{ab}x^a y^b$, it can also be written with respect to point (x_j, r_j) as

$$Q(x,y) = \sum_{\alpha,\beta} Q_{\alpha\beta}^{(x_j,r_j)}(x - x_j)^{\alpha}(y - r_j)^{\beta},$$

where $Q_{\alpha\beta}^{(x_j,r_j)} \in \mathbf{F}_q$. If $Q_{\alpha\beta}^{(x_j,r_j)} = 0$ for $\alpha + \beta < m$, then $Q(x,y)$ interpolates (x_j, r_j) with a multiplicity of m.

– ***Example 7.8*****:** Given a polynomial $Q(x,y) = \sigma(x - \sigma)(y - \sigma^5)^2 + \sigma^3(x - \sigma)^2(y - \sigma^5)^2$, since $Q_{\alpha\beta}^{(\sigma,\sigma^5)} = 0$ for $\alpha + \beta < 3$, it interpolates (σ, σ^5) with a multiplicity of 3.

§ 7.4 Interpolation Based Decoding

– Given $Q(x,y) = \sum_{a,b} Q_{ab} x^a y^b$, the (α,β)-Hasse derivative evaluation at point (x_j, r_j) is

$$D_{\alpha,\beta}\left(Q(x_j, r_j)\right) \triangleq Q_{\alpha\beta}^{(x_j,r_j)} = \sum_{a\geq\alpha, b\geq\beta} Q_{ab} \binom{a}{\alpha}\binom{b}{\beta} x_j^{a-\alpha} r_j^{b-\beta}.$$

– Derivation: Given $Q(x,y) = \sum_{a,b} Q_{ab} x^a y^b$, it can also be written as

$$Q(x,y) = \sum_{\alpha,\beta} Q_{\alpha\beta}^{(x_j,r_j)} (x - x_j)^\alpha (y - r_j)^\beta.$$

Since

$$x^a = (x - x_j + x_j)^a = \sum_{a\geq\alpha} \binom{a}{\alpha} (x - x_j)^\alpha x_j^{a-\alpha},$$

$$y^b = (y - r_j + r_j)^b = \sum_{b\geq\beta} \binom{b}{\beta} (y - r_j)^\beta r_j^{b-\beta},$$

we substitute them into $Q(x,y)$ and get

$$Q(x,y) = \sum_{a,b} Q_{ab} \sum_{a\geq\alpha} \binom{a}{\alpha} (x - x_j)^\alpha x_j^{a-\alpha} \sum_{b\geq\beta} \binom{b}{\beta} (y - r_j)^\beta r_j^{b-\beta}$$

$$= \sum_{\alpha,\beta} \sum_{a\geq\alpha, b\geq\beta} Q_{ab} \binom{a}{\alpha}\binom{b}{\beta} x_j^{a-\alpha} r_j^{b-\beta} (x - x_j)^{a-\alpha} (y - r_j)^{b-\beta}.$$

Therefore, $Q_{\alpha\beta}^{(x_j,r_j)} = \sum_{a\geq\alpha, b\geq\beta} Q_{ab} \binom{a}{\alpha}\binom{b}{\beta} x_j^{a-\alpha} r_j^{b-\beta} \triangleq D_{\alpha,\beta}\left(Q(x_j, r_j)\right)$.

§ 7.4 Interpolation Based Decoding

- **Interpolation Theorem**: If $m\left|\{j: r_j = c_j\}\right| > \deg_{1,k-1} Q$, then $Q\left(x, u(x)\right) = 0$.
- Proof:

① If $Q(x, y)$ interpolates (x_j, c_j) with a multiplicity of m, then

$$Q(x, y) = \sum_{\alpha+\beta\geq m} Q_{\alpha\beta}^{(x_j,c_j)}(x - x_j)^{\alpha}(y - c_j)^{\beta}.$$

② For the message $u(x)$, $u(x_j) = c_j$. Replace c_j by $u(x_j)$,

$$Q(x, y) = \sum_{\alpha+\beta\geq m} Q_{\alpha\beta}^{(x_j,c_j)}(x - x_j)^{\alpha}\left(y - u(x_j)\right)^{\beta}.$$

③ Replace y by $u(x)$,

$$Q(x, u(x)) = \sum_{\alpha+\beta\geq m} Q_{\alpha\beta}^{(x_j,c_j)}(x - x_j)^{\alpha}\left(u(x) - u(x_j)\right)^{\beta}$$

$$= \sum_{\alpha+\beta\geq m} Q_{\alpha\beta}^{(x_j,c_j)}(x - x_j)^{\alpha}\left((x - x_j)\Phi(x)\right)^{\beta}$$

$$= \sum_{\alpha+\beta\geq m} Q_{\alpha\beta}^{(x_j,c_j)}(x - x_j)^{\alpha+\beta}\Phi^{\beta}(x)$$

⇒ When $u(x_j) = c_j$, $(x - x_j)^m | Q(x, u(x))$.

§ 7.4 Interpolation Based Decoding

④ Since $Q(x, y)$ interpolates $(x_0, r_0), (x_1, r_1), \ldots\ldots, (x_{n-1}, r_{n-1})$ with a multiplicity of m, then $(x - x_j)^m | Q(x, u(x))$ holds if $r_j = c_j$ (or $e_j = 0$).

$$\Rightarrow \prod_{j:r_j = c_j} (x - x_j)^m | Q(x, u(x))$$

⑤ In what condition will $Q(x, u(x)) = 0$?

The total number of roots of $Q(x, u(x))$: $m|\{j: r_j = c_j\}|$

Degree of $Q(x, u(x))$: $\deg_x Q + (k-1)\deg_y Q = \deg_{1,k-1} Q$

⑥ Therefore, if $m|\{j: r_j = c_j\}| > \deg_{1,k-1} Q$, then $Q(x, u(x)) = 0$.

– The GS algorithm can correct $n - |\{j: r_j = c_j\}|$ errors.

– The interpolation problem is how to find the smallest $Q(x, y)$.

§ 7.4 Interpolation Based Decoding

– **Monomial ordering**

– The (1, k – 1)-weighted degree of monomial $x^a y^b$:

$$\deg_{1,k-1} x^a y^b = a + (k-1)b$$

– The (1, k – 1)-lexicographic order (ord): ord($x^{a_1}y^{b_1}$) < ord($x^{a_2}y^{b_2}$) if $\deg_{1,k-1} x^{a_1}y^{b_1} < \deg_{1,k-1} x^{a_2}y^{b_2}$, or $\deg_{1,k-1} x^{a_1}y^{b_1} = \deg_{1,k-1} x^{a_2}y^{b_2}$ and $b_1 < b_2$.

– ***Example 7.9*:** In order to decode a (7, 3) RS code, (1, 2)-weighted degree and (1, 2)-lexicographic order of monomial $x^a y^b$ are used.

b \ a	0	1	2	3	4	5	6	7	8	⋯
0	0	1	2	3	4	5	6	7	8	⋯
1	2	3	4	5	6	7	8	9	10	⋯
2	4	5	6	7	8	9	10	11	12	⋯
3	6	7	8	9	10	11	12	13	14	⋯
⋮										

(1, 2)-weighted degree

b \ a	0	1	2	3	4	5	6	7	8	⋯
0	0	1	2	4	6	9	12	16	20	⋯
1	3	5	7	10	13	17	21	⋯		
2	8	11	14	18	22	⋯				
3	15	19	23	⋯						
⋮										

(1, 2)-lexicographic order

§ 7.4 Interpolation Based Decoding

- **Polynomial ordering**
- Any nonzero bivariate polynomial $Q(x, y)$ can be written as

$$Q(x, y) = Q_0 M_0 + Q_1 M_1 + \cdots\cdots + Q_T M_T,$$

where $Q_0, Q_1, \dots\dots, Q_T \in \mathbf{F}_q$ and $Q_T \neq 0$, $M_0 < M_1 < \cdots\cdots < M_T$ are monomials.

- The (1, k – 1)-weighted degree of $Q(x, y)$ is

$$\deg_{1,k-1} Q(x, y) = \deg_{1,k-1} M_T \,.$$

- Leading order (lod) of $Q(x, y)$ is

$$\text{lod}\big(Q(x, y)\big) = \text{ord}(M_T) = T.$$

- ***Example 7.10*****:** Given a polynomial $Q(x, y) = 1 + x^2 + x^2 y + y^2$, applying the (1, 2)-lexicographic order, it has leading monomial $M_T = y^2$. Therefore, $\deg_{1,2}\big(Q(x, y)\big) = \deg_{1,2} y^2 = 4$ and $\text{lod}\big(Q(x, y)\big) = \text{ord}(y^2) = 8$.

- Given two polynomials $Q_1(x, y)$ and $Q_2(x, y)$, $Q_1 \leq Q_2$ if $\text{lod}(Q_1) \leq \text{lod}(Q_2)$.

§ 7.4 Interpolation Based Decoding

- **Decoding parameters**: error-correction capability τ_m and maximum output list size l_m.
- Let

$$S_x(K) = \max\{a: \operatorname{ord}(x^a y^b) \leq K\}$$
$$S_y(K) = \max\{b: \operatorname{ord}(x^a y^b) \leq K\}$$

- The number of iterations in the interpolation process is

$$C = n\binom{m+1}{2}.$$

- Error-correction capability is

$$\tau_m = n - 1 - \left\lfloor \frac{S_x(C)}{m} \right\rfloor.$$

- Maximum output list size is

$$l_m = S_y(C).$$

- ***Example 7.11*****:** To decode the (63, 21) RS code defined over $\mathbf{F}_{64}$, we obtain

m	1	2	3	5	16
τ_m	21	24	25	26	27
l_m	2	3	5	9	28

The BM algorithm can correct $\left\lfloor \frac{n-k}{2} \right\rfloor = 21$ errors.

§ 7.4 Interpolation Based Decoding

– **Koetter's interpolation**

– Hasse deriv. eval.: $D_{\alpha,\beta}\left(Q\left(x_j, r_j\right)\right) = Q_{\alpha\beta}^{(x_j, r_j)} = \sum_{a\geq\alpha, b\geq\beta} Q_{ab}\binom{a}{\alpha}\binom{b}{\beta} x_j^{a-\alpha} r_j^{b-\beta}$

– Two properties of Hasse derivative evaluation

① Linear functional: Let $Q_1, Q_2 \in \mathbf{F}_q[x, y]$, $d_1, d_2 \in \mathbf{F}_q$, then

$$D(d_1Q_1 + d_2Q_2) = d_1D(Q_1) + d_2D(Q_2).$$

② Bilinear Hasse derivative: Let $Q_1, Q_2 \in \mathbf{F}_q[x, y]$, then

$$[Q_1, Q_2]_D \triangleq Q_1D(Q_2) - Q_2D(Q_1).$$

With property ①, we have $D([Q_1, Q_2]_D) = D(Q_1)D(Q_2) - D(Q_2)D(Q_1) = 0$.

– If $\text{lod}(Q_1) > \text{lod}(Q_2)$, $[Q_1, Q_2]_D$ has leading order $\text{lod}(Q_1)$. Therefore, by performing the bilinear Hasse derivative over two polynomials both of which have nonzero evaluations, a polynomial can be reconstructed, which has a zero evaluation.

§ 7.4 Interpolation Based Decoding

– **Koetter's interpolation**

– An iterative polynomial construction algorithm

– Find the minimum (1, k – 1)-weighted degree polynomial $Q(x, y)$ that satisfies

$$Q(x,y) = \min_{\text{lod}(Q)} \left\{ \begin{matrix} Q(x,y) \in \mathbf{F}_q[x,y] | D_{\alpha,\beta}\left(Q\left(x_j, r_j\right)\right) = 0 \text{ for } j = 0,1,\dots,n-1 \\ \text{and } \alpha + \beta < m \end{matrix} \right\}.$$

– Iteratively modify a set of polynomials through all n points with every possible (α, β) pair. With a multiplicity of m, there are $\binom{m+1}{2}$ pairs of (α, β), i.e., (0,0), (0,1), … , $(0, m-1), (1,0), \dots, (1, m-2), \dots, (m-1, 0)$.

– For an (n, k) RS code, there are $C = n\binom{m+1}{2}$ interpolation constraints. This means that we need C iterations to construct the interpolation polynomial $Q(x, y)$.

§ 7.4 Interpolation Based Decoding

– **Koetter's interpolation**

– At the beginning, a group of polynomials are initialized as

$$\mathbf{G} = \{Q_0(x,y), Q_1(x,y), \dots, Q_{l_m}(x,y)\} = \{1, y, y^2, \dots, y^{l_m}\}.$$

– For each point (x_j, r_j) and each (α, β) pair, calculate Hasse derivative for each Q_i, i.e.,

$$\Delta_i = D_{\alpha,\beta}\left(Q_i(x_j, r_j)\right).$$

– Those polynomials with $\Delta_i = 0$ do not need to be updated.

– **Polynomial updating**

Let $i^* = \text{argmin}_i\{Q_i(x,y) | \Delta_i \neq 0\}$ and $Q^*(x,y) = Q_{i^*}(x,y)$.

For those polynomials with $\Delta_{i'} \neq 0$ but $i' \neq i^*$, update them (using Property ② of Hasse derivative) without the leading order increasing as

$$Q_{i'}(x,y) = [Q_{i'}(x,y), Q^*(x,y)]_D = \Delta_{i^*} Q_{i'}(x,y) - \Delta_{i'} Q^*(x,y).$$

For $Q_{i^*}(x,y)$ itself, it is updated with the leading order increasing as

$$Q_{i^*}(x,y) = [xQ^*(x,y), Q^*(x,y)]_D = \Delta_{i^*}(x - x_j)Q^*(x,y).$$

§ 7.4 Interpolation Based Decoding

– Pseudo program of Koetter's interpolation

Koetter's interpolation

```
Initialization: G = {Q_0(x,y), Q_1(x,y), ..., Q_{l_m}(x,y)} = {1, y, y^2, ..., y^{l_m}}
For j = 0 to n – 1 do
    For (α, β) = (0,0) to (m − 1,0) do
        For i = 0 to l_m do
            Δ_i = D_{α,β}(Q_i(x_j, r_j))
        I = {i | Δ_i ≠ 0}
        If I ≠ ∅ do
            i* = argmin_i{Q_i(x,y) | Δ_i ≠ 0}
           Q*(x,y) = Q_{i*}(x,y)
           For i' ∈ I do
              If i' ≠ i* do
                 Q_{i'}(x,y) = Δ_{i*} Q_{i'}(x,y) − Δ_{i'} Q*(x,y)
              Else if i' = i* do
                 Q_{i*}(x,y) = Δ_{i*}(x − x_j) Q*(x,y)
Output: Q(x,y) = min{Q_0(x,y), Q_1(x,y), ..., Q_{l_m}(x,y)}
```

(Lines 12–14:) Use property ② of Hasse derivative to update polynomials.

§ 7.4 Interpolation Based Decoding

– ***Example 7.12***: Given the received word generated in ***Example 7.6***, i.e.,

$$\bar{r} = (\sigma^5, \sigma^4, \sigma^3, \sigma^0, \sigma^4, \sigma^2, \sigma^5)$$

The interpolation points are $(\sigma^0, \sigma^5), (\sigma^1, \sigma^4), (\sigma^2, \sigma^3), (\sigma^3, \sigma^0)$, (σ^4, σ^4), (σ^5, σ^2), (σ^6, σ^5).

Let $m = 1$, then $C = 7$ and $l_m = 1$. Initialize $\mathbf{G} = \{1, y\}$. Running Koetter's interpolation, we obtain

j	(α, β)	Δ_0	Δ_1	$\text{lod}(Q_0)$	$\text{lod}(Q_1)$	i^*	$\mathbf{G}$
0	(0, 0)	σ^0	σ^5	0	3	0	$\{1 + x, \sigma^5 + y\}$
1	(0, 0)	σ^3	1	1	3	0	$\{\sigma^4 + \sigma^6 x + \sigma^3 x^2, \sigma^3 + x + \sigma^3 y\}$
2	(0, 0)	σ^6	σ	2	3	0	$\{\sigma^5 + \sigma x + x^2 + \sigma^2 x^3, \sigma^3 + \sigma^2 x + \sigma^4 x^2 + \sigma^2 y\}$
3	(0, 0)	σ	σ^3	4	3	1	$\{\sigma^2 + \sigma^6 x + \sigma^2 x^2 + \sigma^5 x^3 + \sigma^3 y, \sigma^2 + \sigma^5 x + \sigma^2 x^2 + x^3 + (\sigma + \sigma^5 x)y\}$
4	(0, 0)	σ^4	σ^4	4	5	0	$\{\sigma^3 + \sigma^2 x + \sigma^2 x^4 + (\sigma^4 + x)y, \sigma^5 x + \sigma x^3 + (\sigma^4 + \sigma^2 x)y\}$
5	(0, 0)	σ^2	σ^4	6	5	1	$\{1 + \sigma^2 x + \sigma^3 x^3 + \sigma^6 x^4 + \sigma^5 y, x + \sigma^2 x^2 + \sigma^3 x^3 + \sigma^5 x^4 + (\sigma^6 + \sigma^2 x + \sigma^6 x^2)y\}$
6	(0, 0)	σ^6	0	6	7	0	$\{\sigma^5 + \sigma^2 x + \sigma x^2 + \sigma x^3 + \sigma x^4 + \sigma^5 x^5 + (\sigma^3 + \sigma^4 x)y,$ $x + \sigma^2 x^2 + \sigma^3 x^3 + \sigma^5 x^4 + (\sigma^6 + \sigma^2 x + \sigma^6 x^2)y\}$

Since $\text{lod}\big(Q_0(x, y)\big) = 9, \text{lod}\big(Q_1(x, y)\big) = 7$, the interpolation polynomial $Q(x, y) = Q_1(x, y) = x + \sigma^2 x^2 + \sigma^3 x^3 + \sigma^5 x^4 + (\sigma^6 + \sigma^2 x + \sigma^6 x^2)y$.

§ 7.4 Interpolation Based Decoding

– **Factorization**: Recursive coefficients search algorithm

– Let $f(x) = f_0 + f_1 x + f_2 x^2 + \cdots + f_{k-1} x^{k-1}$ denote a y-root of $Q(x, y)$. Factorization can be realized through recursively deducing $f_0, f_1, f_2, \ldots, f_{k-1}$ one by one.

– For any bivariate polynomial, if h is the highest degree such that $x^h | Q(x, y)$, we define

$$Q'(x, y) = \frac{Q(x, y)}{x^h}.$$

– Denote $Q_0(x, y) = Q'(x, y)$, we define the recursively updated polynomial $Q_s(x, y)$ $(s \geq 1)$ as

$$Q_s(x, y) = Q'_{s-1}(x, xy + f_{s-1}),$$

where f_{s-1} is the roots of $Q_{s-1}(0, y) = 0$.

– Pseudo program of factorization

Factorization
1: Initialization: $Q_0(x, y) = Q'(x, y), s = 0$
2: Find roots f_s of $Q_s(0, y) = 0$
3: For each f_s, perform $Q_{s+1}(x, y) = Q'_s(x, xy + f_s)$
4: $s = s + 1$
5: If $s < k$, go to Step **2**. If $s = k$ and $Q_s(x, 0) \neq 0$, stop this deduction root. If $s = k$ and $Q_s(x, 0) = 0$, trace the deduction root to find $f_{s-1}, \ldots, f_1, f_0$.

§ 7.4 Interpolation Based Decoding

– ***Example 7.13*:** Given the interpolation polynomial $Q(x, y)$ obtained in ***Example 7.12***, initialize $Q_0(x, y) = Q'(x, y) = (x + \sigma^2 x^2 + \sigma^3 x^3 + \sigma^5 x^4) + (\sigma^6 + \sigma^2 x + \sigma^6 x^2)y$ and $s = 0$. Then, $Q_0(0, y) = \sigma^6 y$ and $f_0 = 0$ is the root of $Q_0(0, y) = 0$.

Update $Q_1(x, y) = Q'_0(x, xy + f_0) = (1 + \sigma^2 x + \sigma^3 x^2 + \sigma^5 x^3) + (\sigma^6 + \sigma^2 x + \sigma^6 x^2)y$ and $s = s + 1 = 1$.

As $s < k$, go to Step **2**.

Then, $Q_1(0, y) = 1 + \sigma^6 y$ and $f_1 = \sigma$ is the root of $Q_1(0, y) = 0$.

Update $Q_2(x, y) = Q'_1(x, xy + f_1) = (\sigma^5 + \sigma x + \sigma^5 x^2) + (\sigma^6 + \sigma^2 x + \sigma^6 x^2)y$ and $s = s + 1 = 2$.

As $s < k$, go to Step **2**.

Then, $Q_2(0, y) = \sigma^5 + \sigma^6 y$ and $f_2 = \sigma^6$ is the root of $Q_2(0, y) = 0$.

Update $Q_3(x, y) = Q'_2(x, xy + f_2) = (\sigma^6 + \sigma^2 x + \sigma^6 x^2)y$ and $s = s + 1 = 3$.

As $s = k$ and $Q_3(x, 0) = 0$, trace this deduction root to find the coefficients $f_0 = 0$, $f_1 = \sigma$, $f_2 = \sigma^6$. Therefore, the factorization output is $f(x) = \sigma x + \sigma^6 x^2$. According to ***Example 7.4*** $f(x)$ matches the transmitted message polynomial $u(x)$.

§ 7.4 Interpolation Based Decoding

– Performance of the (63, 21) RS code over the AWGN channel using BPSK