# Algebraic-Geometric Code and Modernised Algebraic Decoding


- Dr. Li Chen
  - Lecturer, School of Information Science and Technology, Sun Yat-sen University
  - BSc, MSc, PhD, MIEEE
  - Website: http://sist.sysu.edu.cn/~chenli

# Personal Background

- Education and employment
  - 2003, BSc in Applied Physics, Jinan University, China
  - 2004, MSc in Communications and Signal Processing, Newcastle University, UK
  - 2008, PhD in Mobile Communications, Newcastle University, Supervisor: Prof. R. A. Carrasco (IET Fellow)
  - 2007 – 2010, Research Associate, Newcastle University, engaged with an EPSRC project.
  - 2010 -- .., Lecturer, Sun Yat-sen University

- Research Interests
  - Information theory and channel coding
  - Cooperative system

# Outline

- Part Ⅰ - Algebraic-geometric codes
  - Construction of Hermitian Codes
  - Algebraic soft decoding of Hermitian codes
  - Performance evaluation (Hermitian vs. RS)
  - (Made in UK)
- Part Ⅱ - Modernised algebraic decoding
  - Challenges → Inspiration
  - Modernisation: Progressive algebraic soft decoding (PASD)
  - Complexity reduction and performance evaluation
  - (Made in China)
- Conclusions and future work

# I.Construction of Hermitian Codes

- Hermitian Curve: $H_w(x, y, z) = x^{w+1} + y^w z + yz^w$
  - Affine component: $H_w(x, y, 1) = x^{w+1} + y^w + y$ – used for code construction!
- Size of GF($q$) decides the degree of the curve: $w = \sqrt{q}$
- Genus of the curve: $g = w(w-1)/2$
- Designed distance of a ($n$, $k$) Hermitian code: $d^* = n - k - g + 1$
- Size of the code: number of affine points $p_i = (x_i, y_i)$, $|p_i| = w^3$ ($> q$)
- Codes on fields

| Paras \ GF($q$) | GF(4) | GF(16) | GF(64) | GF(256) |
|---|---|---|---|---|
| deg | 3 | 5 | 9 | 17 |
| $g$ | 1 | 6 | 28 | 120 |
| $n$ | 8 | 64 | 512 | 4096 |

# I.Construction of Hermitian Codes

- Point of infinity $p_\infty$: for points that we can find in $H_w(1, y, z)$, $H_w(x, 1, z)$ and $H_w(x, y, 1)$, the one with the form of $(x_i, y_i, 0)$.
  - Variables $x$, $y$, $z$ have a pole order (or weights) at $p_\infty$, $x$ – $w$, $y$ – $w$+1, $z$ -- ? (depends on $k$).

- Affine points $p_i$: points on an affine component. E.g. for $H_w(x, y, 1)$, $p_i$ satisfies $H_w(x_i, y_i, 1) = 0$.

- Pole basis $L_w$: a set of rational functions $\Phi_\alpha$ with increasing pole orders
  - Curve $H_2$ has $L_2 = \{1, x, y, x^2, xy, y^2, x^2y, xy^2, y^3, x^2y^2, xy^3, y^4, \ldots\}$

    **1**

  - Curve $H_4$ has $L_4 = \{1, x, y, x^2, xy, y^2, x^3, x^2y, xy^2, y^3, x^4, x^3y, x^2y^2, xy^3, y^4, x^4y, x^3y^2, x^2y^3, xy^4, y^5, \ldots\}$

    **1, 2, 3** **6, 7** **11**

- Zero basis $Z_{w,pi}$: a set of rational functions $\psi_{w,pi}$ with increasing zero orders at $p_i$.

# I.Construction of Hermitian Codes

- For a Hermitian code defined on the curve $H_w$:
  - Find out $n$ affine points on the curve – decide the length of the code
  - Select the first $k$ monomials in $L_w$ – decide the dimension of the code
  - With information symbols $(u_0, u_1, \ldots, u_{k-1}) \in \mathrm{GF}(q)$, the message polynomial can be written as:

$$u(x, y) = u_0\Phi_0 + u_1\Phi_1 + \ldots + u_{k-1}\Phi_{k-1}$$

  - And the codeword is generated by:

$$(c_0, c_1, \ldots, c_{n-1}) = (u(p_0), u(p_1), \ldots, u(p_{n-1}))$$

- Example: Construct a (8, 4) Hermitian code defined over $\mathrm{GF}(2^2)$
  - Curve: $H_2 = x^3 + y^2 + y$
  - Affine points $p_0 = (0, 0)$, $p_1 = (0, 1)$, $p_2 = (1, \sigma)$, $p_3 = (1, \sigma^2)$, $p_4 = (\sigma, \sigma)$, $p_5 = (\sigma, \sigma^2)$, $p_6 = (\sigma^2, \sigma)$, $p_7 = (\sigma^2, \sigma^2)$.
  - Information symbols 1, $\sigma$, 1, $\sigma^2$, and message polynomial $u(x, y) = 1 + \sigma x + y + \sigma^2 x^2$.
  - Codeword $(c_0, c_1, c_2, c_3, c_4, c_5, c_6, c_7) = (1, 0, \sigma, \sigma^2, \sigma, \sigma^2, \sigma^2, \sigma)$.

# I. A Comparison with RS Codes

| Properties \ Codes | $(n, k)$ RS code | $(n, k)$ Hermitian code |
|---|---|---|
| Algebraic affine curves | $y = 0$ | $x^{w+1} + y^w + y = 0$ |
| Pole basis | $1, x, x^2, x^3, \ldots$ | $1, x, y, x^2, xy, y^2, \ldots, x^w y, x^{w-1}y^2, \ldots, xy^w, y^{w+1}, \ldots$ |
| Affine points $(p)$ | $x_0, x_1, x_2, \ldots, x_{n-1}$ | $(x_0, y_0), (x_1, y_1), (x_2, y_2), \ldots, (x_{n-1}, y_{n-1})$ |
| Transmitted message polynomial $(u)$ | $u(x) = u_0 + u_1 x + u_2 x^2 + \ldots + u_{k-1}x^{k-1}$ | $u(x, y) = u_0 + u_1\phi_1 + u_2\phi_2 + \ldots + u_{k-1}\phi_{k-1}$ |
| Codeword $(\bar{c})$ | $(c_0, c_1, \ldots, c_{n-1}) = (u(x_0), u(x_1), \ldots, u(x_{n-1}))$ | $(c_0, c_1, \ldots, c_{n-1}) = (u(p_0), u(p_1), \ldots, u(p_{n-1}))$ |

# I. A Comparison with RS codes

- Advantage of AG codes: larger codes can be constructed from the same finite field as RS codes, resulting better error-correction capability;

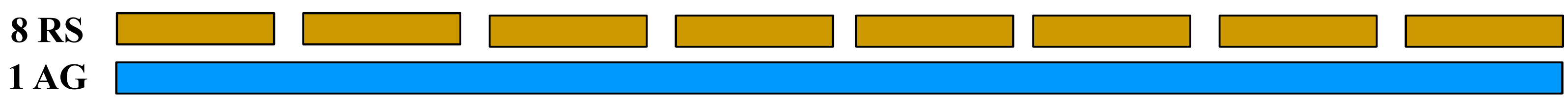
- Example, over GF(64)

| Rate 0.3 | | Rate 0.56 | |
|---|---|---|---|
| Herm (512, 153) | RS (63, 19) | Herm (512, 289) | RS (63, 35) |
| $d^* = 332$ | $d = 45$ | $d^* = 196$ | $d = 29$ |
| $\tau = 165$ | $\tau = 22$ | $\tau = 97$ | $\tau = 14$ |
| 990 bits | 132 bits | 582 bits | 84 bits |

- Disadvantage of AG codes: It is not a Maximum Distance Separable (MDS) code. Very high rate AG codes will be left with marginal error-correction capability.

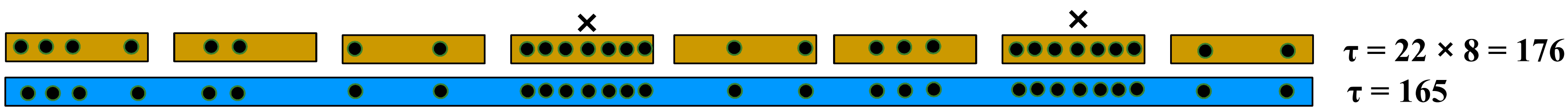
# I. A Comparison with RS codes

- AG vs. concatenated RS (512 ≈ 8 × 63)

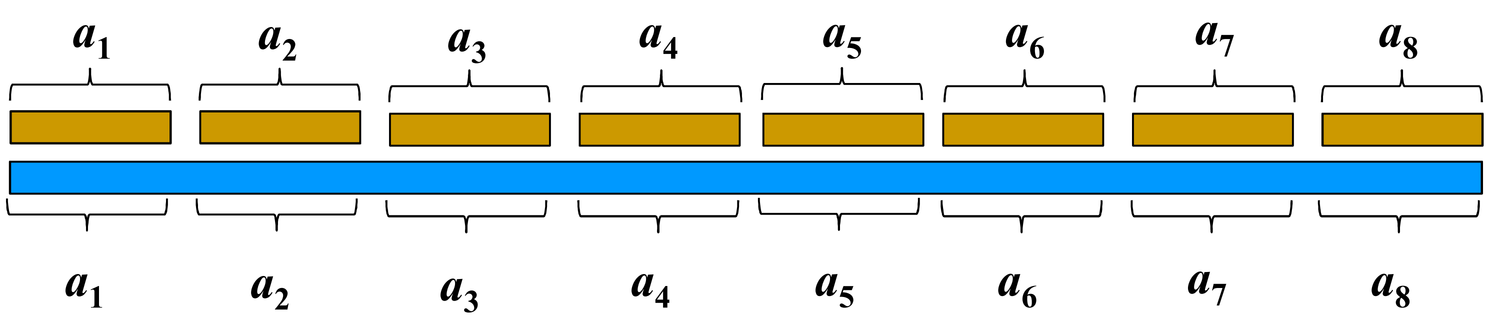



- Complexity: $\boldsymbol{O}(n^h)$

- Distribution of errors



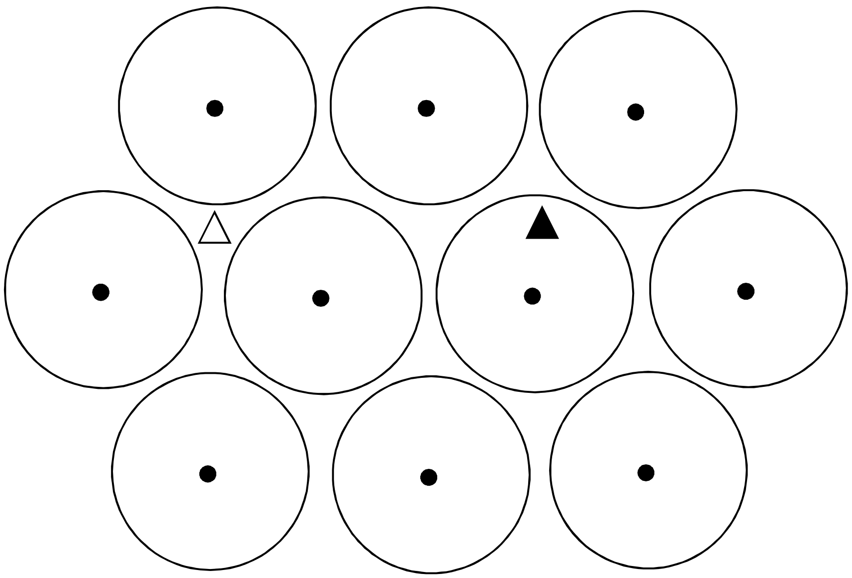

- Diversity on codes



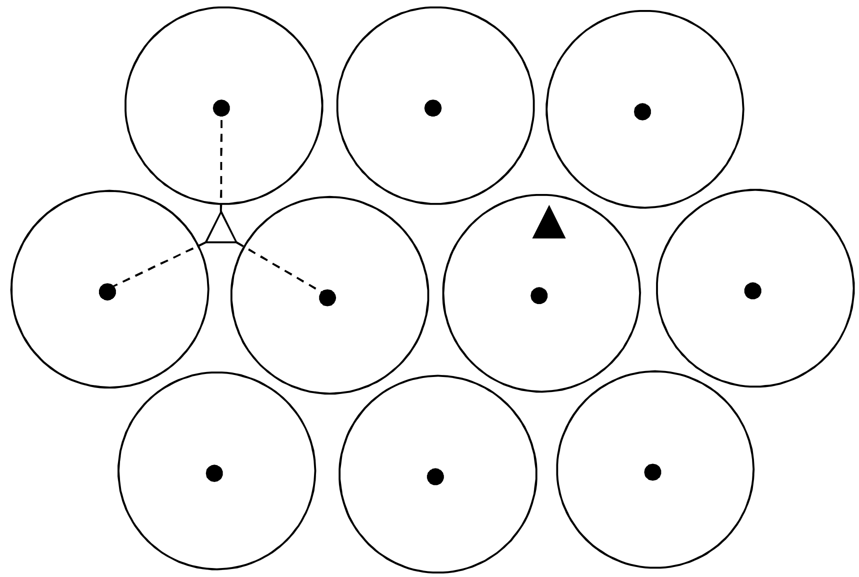
# I.Overview of the algebraic decoding

- Decoding philosophy evolution

**Unique decoding** → **List decoding**



The Berlekamp-Massey algorithm
The Welch-Berlekamp algorithm
The Sakata algorithm with majority voting



The Guruswami-Sudan algorithm (Hard-decision)
The Koetter-Vardy algorithm (Soft-decision)
[Guruswami99], [Koetter03]

# I.Overview of the algebraic decoding

- Key processes: Interpolation (construct $Q(x, y, z)$) + Factorisation (find out $u(x, y)$)
- From hard-decision decoding to soft-decision decoding (GS → KV)

**Hard-decision received word:** $\overline{R} = (r_0, r_1, \ldots, r_{n-1})$
**Interpolated points:** $(p_0, r_0), (p_1, r_1), \ldots, (p_{n-1}, r_{n-1})$
**With certain multiplicity value $m$, perform:**

**Interpolation $Q(x, y, z)$** → **Factorisation $u(x, y)$**

**Soft-decision reliability matrix Π (→ M)**

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Encoding** | $p_0$ | $p_1$ | $p_2$ | $p_3$ | $p_4$ | $p_5$ | $p_6$ | $p_7$ |
| | $c_0$ | $c_1$ | $c_2$ | $c_3$ | $c_4$ | $c_5$ | $c_6$ | $c_7$ |
| **Channel** | $r_0$ | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $r_5$ | $r_6$ | $r_7$ |

$$\Pi = \begin{bmatrix} \boxed{0.96} & \boxed{0.21} & 0.01 & \boxed{0.46} & 0.00 & 0.00 & \boxed{0.69} & 0.00 \\ 0.00 & 0.02 & 0.00 & \boxed{0.53} & \boxed{0.90} & 0.03 & 0.00 & 0.01 \\ 0.03 & \boxed{0.74} & 0.03 & 0.01 & \boxed{0.10} & 0.02 & \boxed{0.28} & \boxed{0.99} \\ 0.01 & 0.03 & \boxed{0.94} & 0.00 & 0.00 & \boxed{0.95} & 0.03 & 0.00 \end{bmatrix} \begin{matrix} 0 \\ 1 \\ \sigma \\ \sigma^2 \end{matrix}$$

$\boxed{\phantom{0.00}}$ **where a multiplicity value $m_{ij}$ was assigned to the unit**

$(p_0, 0)$ $(p_1, 0)$ $(p_2, \sigma^2)$ $(p_3, 0)$ $(p_4, 1)$ $(p_5, \sigma^2)$ $(p_6, 0)$ $(p_7, \sigma)$
$(p_1, \sigma)$ $(p_3, 1)$ $(p_4, \sigma)$ $(p_6, \sigma)$

# I. Algebraic soft decoding of Hermitian codes

- From RS to Hermitian: **[Chen09], [Lee10]**
  - Bivariate monomials (polynomials) → trivariate monimials (polynomials)
  - Define the interpolated zero conditions
    - Calculate the corresponding coefficients of a Hermitian curve
  - Validity of the algorithm
  - Optimal performance bound
  - Complexity reduction methods

# I. Trivariate monomials (Polynomials)

- For a code defined on the curve $H_w = x^{w+1} + y^w + y$,
  - monomial $x^i y^j z^k$, $0 \le i \le w$, $j \ge 0$ and $k \ge 0$
  - Decoding a $(n, k)$ Hermitian codes, $\deg_w(z) = \deg_w(\Phi_{k-1})$
  - $\deg_w(x^i y^j z^k) = iw + j(w+1) + k\deg_w(z)$
  - For to monomials $x^{i1} y^{j1} z^{k1}$ and $x^{i2} y^{j2} z^{k2}$

    $$x^{i1} y^{j1} z^{k1} < x^{i2} y^{j2} z^{k2}$$

    if $\deg_w(x^{i1} y^{j1} z^{k1}) < \deg_w(x^{i2} y^{j2} z^{k2})$, or $\deg_w(x^{i1} y^{j1} z^{k1}) = \deg_w(x^{i2} y^{j2} z^{k2})$ and $k1 < k2$.
  - A lexicographic order can be assigned to monomials.

- Polynomials $Q(x, y, z) = \sum_{a,b \in N} Q_{ab} \phi_a(x, y) z^b$ , $Q_{ab} \in \mathrm{GF}(q)$
  - Identify the maximal monomial in $Q(x, y, z)$ as $\Phi_{a'} z^{b'}$, then $\deg_w(Q) = \deg_w(\Phi_{a'} z^{b'})$
  - Leading order, $\mathrm{lod}(Q) = \mathrm{ord}(\Phi_{a'} z^{b'})$

- $N_w(\delta) = |\{\phi_a z^b : \deg_w(\phi_a z^b) \le \delta, (a, b, \delta) \in N\}|$ **Define the number of monomials**

  $\Delta_w(v) = \min\{\delta : N_w(\delta) > v, v \in N\}$ **Define the weighted degree of monomials**

# I. Define the Interpolated Zero Conditions

- To interpolate unit $(p_i, r_i)$ (or $(x_i, y_i, r_i)$)
- Recall the zero basis $Z_{w,pi}$ with rational functions $\psi_{pi,\alpha}$ as:

$$\psi_{p_i,\alpha} = \psi_{p_i,\lambda+(w+1)\delta} = (x-x_i)^{\lambda}[(y-y_i)-x_i^{\,w}(x-x_i)]^{\delta}, (0 \le \lambda \le w, \delta \ge 0)$$

- Zero condition with multiplicity $m$ for polynomial $Q(x, y, z) = \sum_{a,b\in N} Q_{ab}\phi_a(x,y)z^b$
  - It can be written as: $Q(x,y,z) = \sum_{\alpha,\beta\in N} Q_{\alpha\beta}^{(p_i,r_i)}\psi_{p_i,\alpha}(z-r_i)^{\beta}$
  - $Q_{\alpha\beta}^{(p_i,r_r)} = 0$ for $\alpha + \beta < m$.
- Since $\phi_a = \sum_{\alpha\in N}\gamma_{a,p_i,\alpha}\psi_{p_i,\alpha}$ and $z^b = \sum_{\beta\le b}\binom{b}{\beta} r_i^{\,b-\beta}(z-r_i)^{\beta}$

$$Q_{\alpha\beta}^{(p_i,r_i)} = \sum_{a,b\ge\beta} Q_{ab}\binom{b}{\beta}\gamma_{a,p_i,\alpha} r_i^{\,b-\beta} \qquad \text{[Nielsen01]}$$

**A key parameter for determining the polynomial's zero condition!**

# Ⅰ.Calculate the Corresponding Coefficients

- Lemma: $\phi_a = \sum_{\alpha \in N} \gamma_{a,p_i,\alpha} \psi_{p_i,\alpha} \longleftrightarrow \psi_{p_i,\alpha} = \sum_{a \in N} \zeta_a \phi_a$ , $\psi_{p_i,\alpha} = \sum_{a \in N, a<L} \zeta_a \phi_a + \phi_L$.

- Recursive corresponding coefficient search algorithm **[Chen08]**

*Algorithm A:* **Determining the corresponding coefficients $\gamma_{a,p_i,\alpha}$ between a pole basis monomial $\phi_a$ and zero basis functions $\psi_{p_i,\alpha}$.**

**Step 1: Initialise all corresponding coefficients $\gamma_{a,p_i,\alpha} = 0$;**

**Step 2: Find the zero basis function $\psi_{p_i,\alpha}$ with $LM(\psi_{p_i,\alpha})$ =**

**$\phi_a$, and let $\gamma_{a,p_i,\alpha} = 1$;**

**Step 3: Initialise function $\hat{\psi} = \psi_{p_i,\alpha}$;**

**Step 4: While $(\hat{\psi} \neq \phi_a)$ {**

**Step 5: Find the second largest pole basis monomial $\psi_{L-1}$ with coefficient $\zeta_{L-1}$ in $\hat{\psi}$;**

**Step 6: In $Z_{w,p_i}$, find a zero basis function $\psi_{p_i,\alpha}$ whose leading monomial $LM(\psi_{p_i,\alpha}) = \phi_{L-1}$, and let the corresponding coefficient $\gamma_{a,p_i,\alpha} = \zeta_{L-1}$;**

**Step 7: Update $\hat{\psi} = \hat{\psi} + \gamma_{a,p_i,\alpha}\psi_{p_i,\alpha}$;**

**}**

# I. Validity of the Algorithm

- **<u>Condition 1:</u>** From the perspective of solving a linear equation group

$$N_w(\delta) > C_M$$

Freedom (Nr of coefficients) Constraints

- **<u>Condition 2:</u>** From the perspective of solving equation $Q(x, y, u) = 0$

$$S_M(\bar{c}) > \deg_w(Q(x, y, z))$$

Total zero order of $Q$ Pole order of $Q$

- **Theorem 2:** Given the multiplicity matrix **M** and the resulting interpolated polynomial $Q(x, y, z)$, if the codeword score $S_M(\bar{c})$ is large enough such that:

$$S_M(\bar{c}) > \deg_w(Q(x, y, z))$$

message polynomial $u$ can be found out by factorising $Q$ as: $z - u \mid Q(x, y, z)$ or $Q(x, y, u) = 0$. → <u>This gives a tight condition of successful list decoding!!!</u>

[Chen09]

# I. Prove the Validity of the Algorithm

- A corollary that can embrace both of the successful decoding conditions.

  **Corollary 3:** Message polynomial f can be found out by $z - u \mid Q(x, y, z)$ if

  $$S_M(\bar{c}) > \Delta_w(C_M)$$

  Since $\Delta_w(C_M)$ guarantees $N_w(\delta) > C_M$ (Condition 1 is met!)

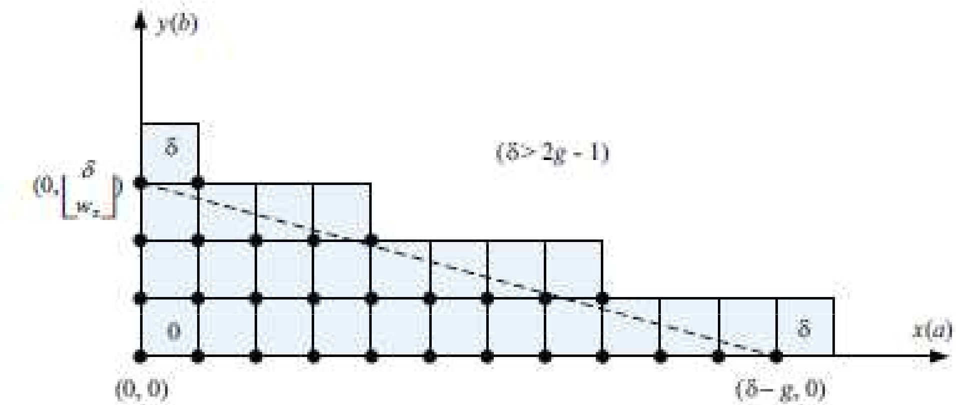
  Since $\deg_w(Q(x, y, z)) \leq \Delta_w(C_M)$, if $S_M(\bar{c}) > \Delta_w(C_M)$, $S_M(\bar{c}) > \deg_w(Q)$ (Condition 2 is met!)

  **This can be seen later.**

- Remark: Solving the linear polynomial group does not give a tight bound on successful list decoding, but solving the polynomial $Q(x, y, u) = 0$ does!

# I.Optimal Performance Bound

- **Corollary 4:** Let $w_z = \deg_w(\Phi_{k-1})$, $N_w(\delta) > \delta(\delta - g)/2w_z$ given $\delta > 2g - 1$. And $N_w(\delta) = \delta^2/2w_z$ with $\delta \rightarrow \infty$.




- With $l \rightarrow \infty$, algebraic soft decoding algorithm's asymptotic optimal performance can be achieved.

  $l \rightarrow \infty$, $C_M \rightarrow \infty$ and $\Delta_w(C_M) \rightarrow \infty$, it results $\Delta_w(C_M) \cong \sqrt{2w_z C_M}$

- Corollary 3 ($S_M(\overline{c}) > \Delta_w$ (CM) ) can be interpreted as:

$$\sum_{j=0}^{n-1} \hat{m}_{i,j} > \sqrt{w_z \sum_{i=0}^{q-1} \sum_{j=0}^{n-1} m_{i,j}(m_{i,j}+1)}.$$

[Chen09]

# I.Optimal Performance Bound

- Asymptotic condition (when $C_M \rightarrow \infty$): $\frac{\pi_{i,j}}{n} = \frac{m_{i,j}}{s}$

- We could further have

$$\frac{s}{n}\sum_{j=0}^{n-1}\widehat{\pi}_{i,j} > \frac{s}{n}\sqrt{w_z\sum_{i=0}^{q-1}\sum_{j=0}^{n-1}\pi_{i,j}(\pi_{i,j}+\frac{n}{s})}.$$

- Since with $s \rightarrow \infty$, $n/s \rightarrow 0$ and

$$\sum_{j=0}^{n-1}\widehat{\pi}_{i,j} > \sqrt{w_z\sum_{i=0}^{q-1}\sum_{j=0}^{n-1}\pi_{i,j}^2}.$$

**In KV decoding of RS codes, $w_z$ is replaced by $k$ - 1**

- The performance of the KV algorithm is bounded by the quality of the received information Π.

- Had the quality of Π been improved, optimal performance bound can be enhanced.

**[El-Khamy06]**

# I. Complexity Reduction Methods

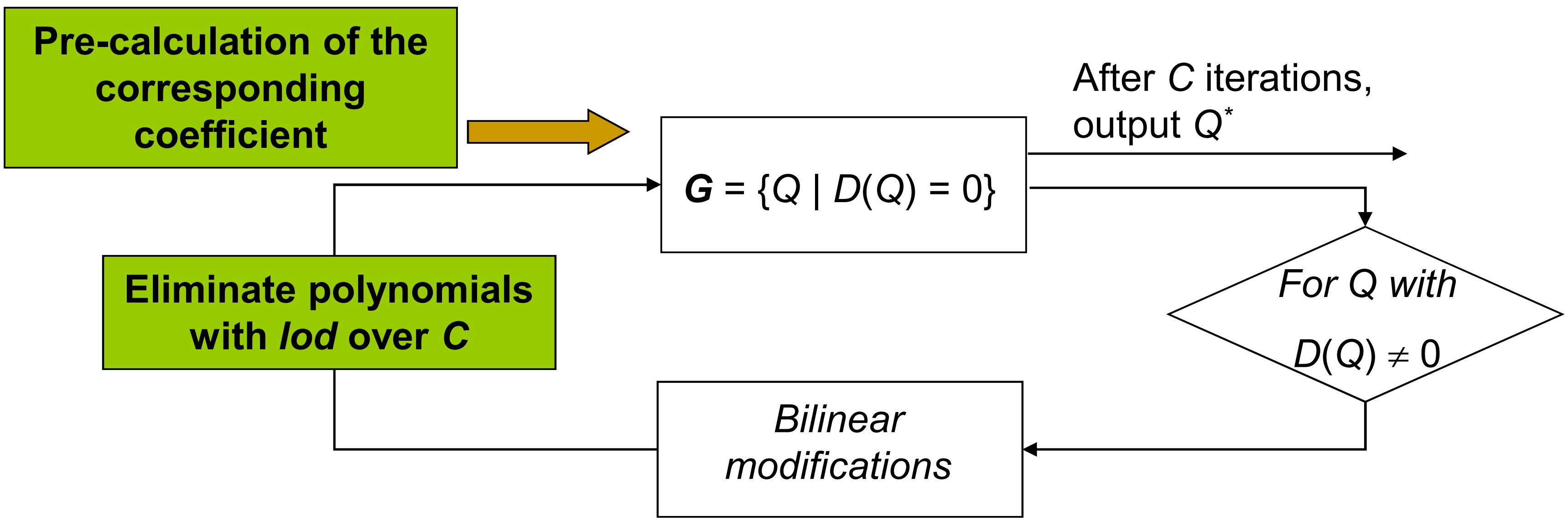
- Modified reliability transform algorithm (introducing a stopping criterion) **[Chen09]**
  - In KV, reliability transform is stopped once a predefined s $=\sum_{i,j} m_{i,j}$ is met.
  - Reliability transform is stopped once a predefined output list size *l* is met.

- Pre-calculation of the corresponding coefficients **[Chen08]**
  - Determine $\gamma_{a,p_i,\alpha}$

- Elimination of the unnecessary polynomials in the group **[Chen07]**
  - Eliminate polynomials with lod($Q$) > $C_M$

# I.Complexity reducing interpolation

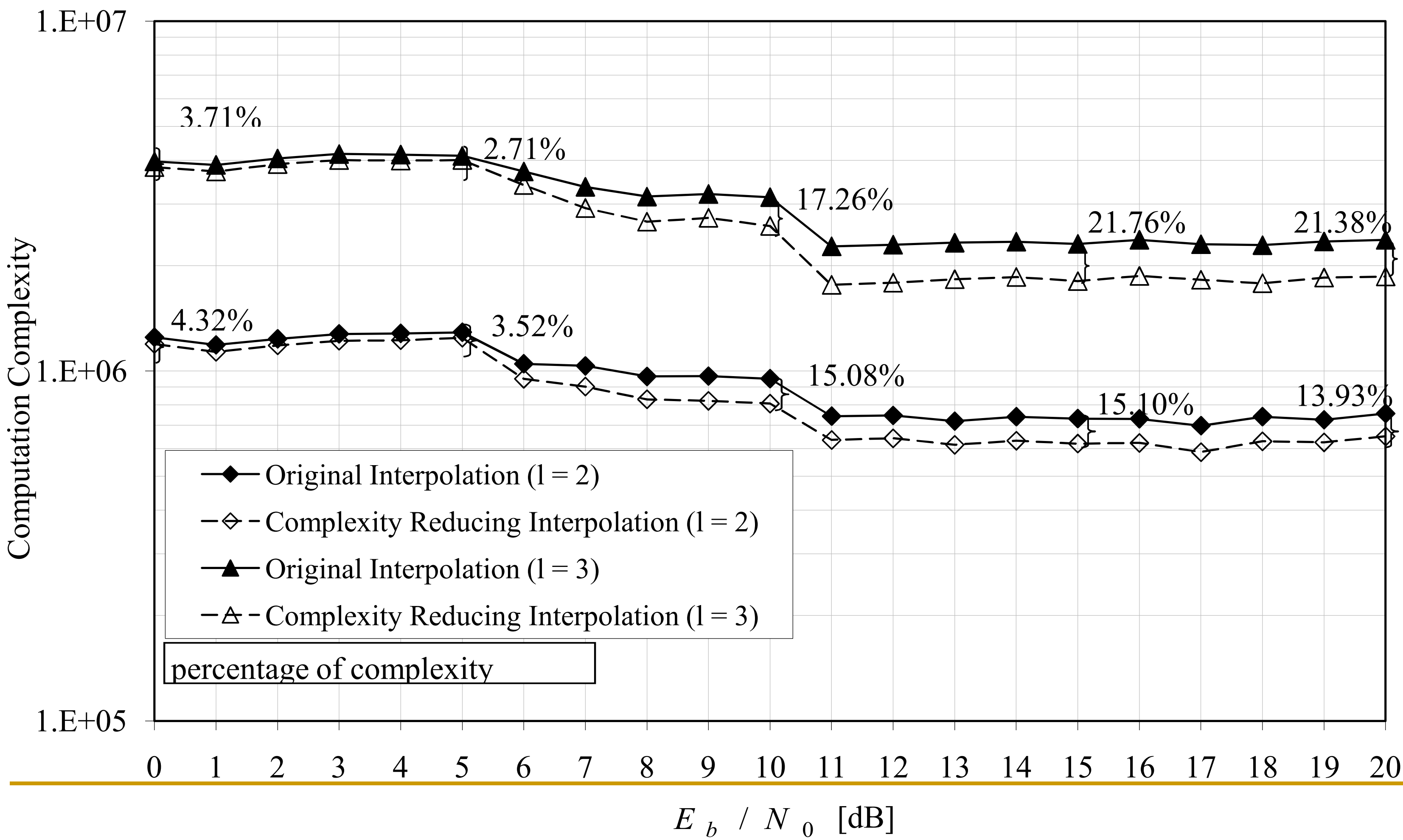
- Pre-calculation of the corresponding coefficients and elimination of the unnecessary polynomials




**In the end, the minimal polynomial *Q* in group *G* is chosen!**

# I.Complexity reducing interpolation

The (64, 19) Hermitian code



# I. Arising Awareness

- Why Condition 1 ($N_w(\delta) > C_M$) is NOT a tight bound?

- Since lod($Q^*$) ≤ $C_M$, if $\deg_w(Q^*) = \delta^*$, then

$$N_w(\delta^*) \leq C_M \longleftrightarrow N_w(\delta) > C_M$$

- $N_w(\delta) > C_M$ is the successful decoding criterion w.r.t. the polynomial group $G$. However, the minimal polynomial in $G$ does not meet this condition.

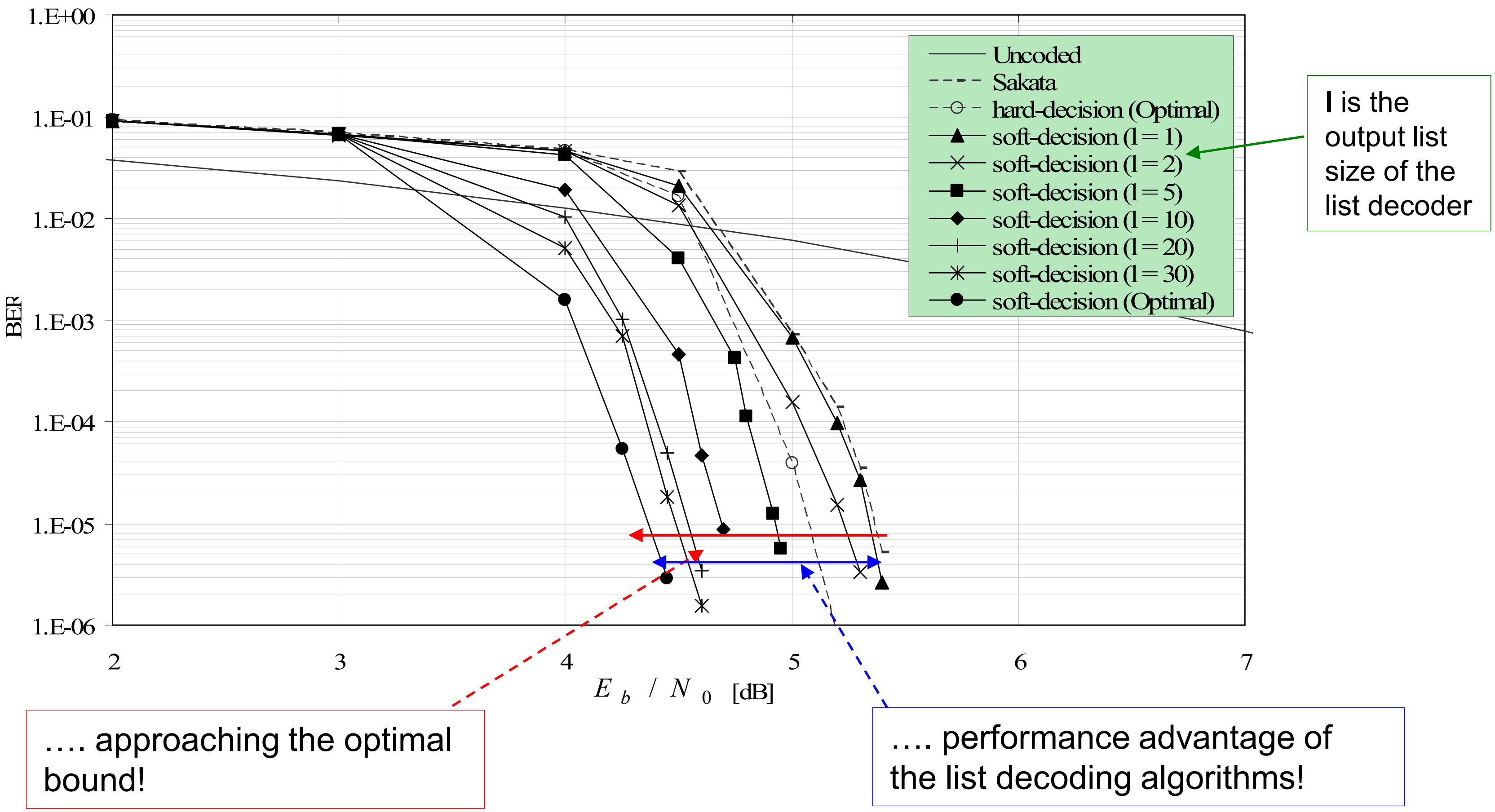
- To access the decoding performance, only Condition 2 gives a tight bound:

$$S_M(\bar{c}) > \deg_w(Q(x, y, z))$$

- Since $\deg_w(Q(x, y, z)) \leq \Delta_w(C_M)$, without performing the interpolation process, the theoretical assessment (e.g. $S_M(\bar{c}) > \Delta_w(C_M)$) produces a relatively negative results.

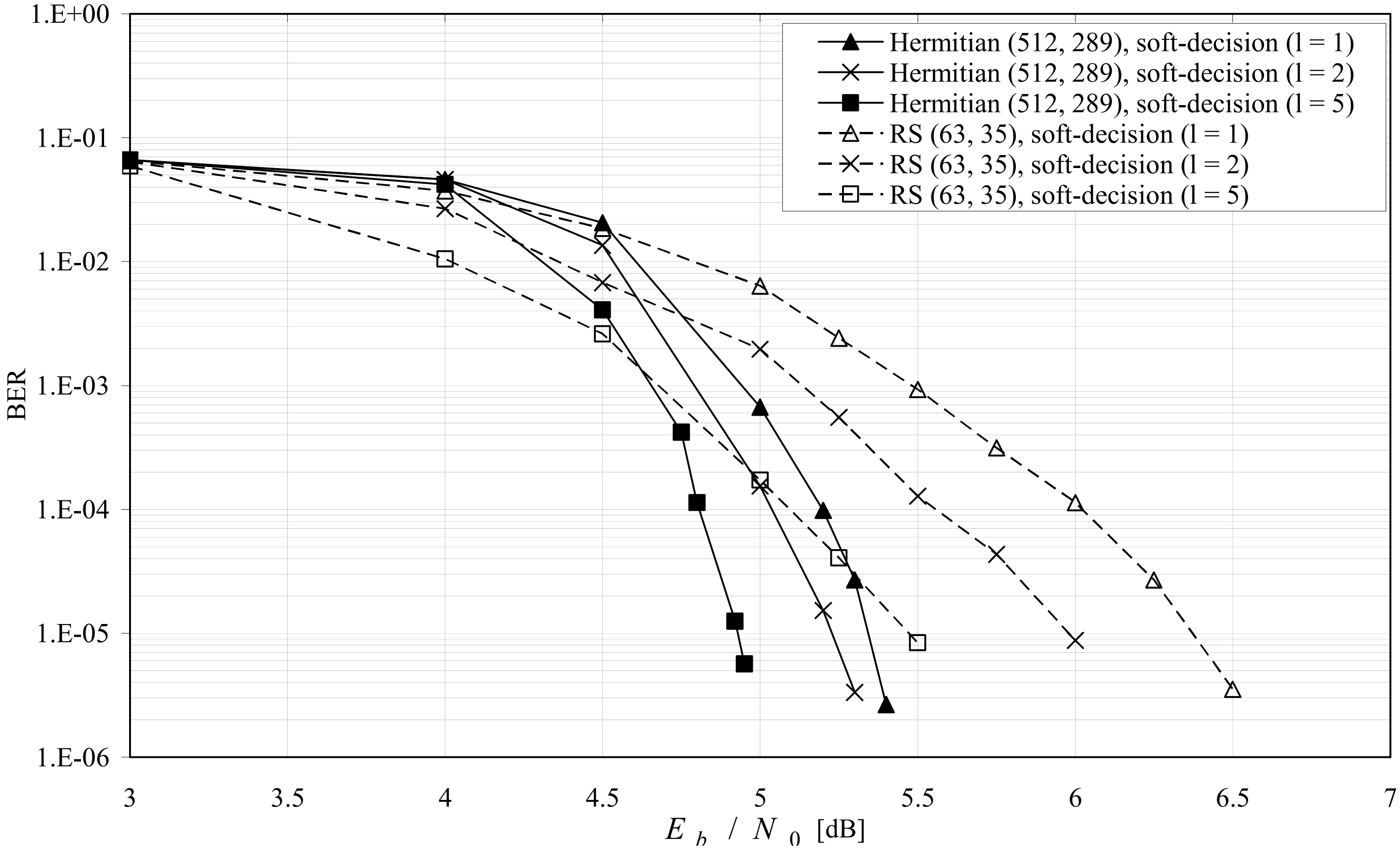
# I.Performance Evaluation

Hermitian code (512, 289) over AWGN channel



# I.Hermitian code ~ RS code

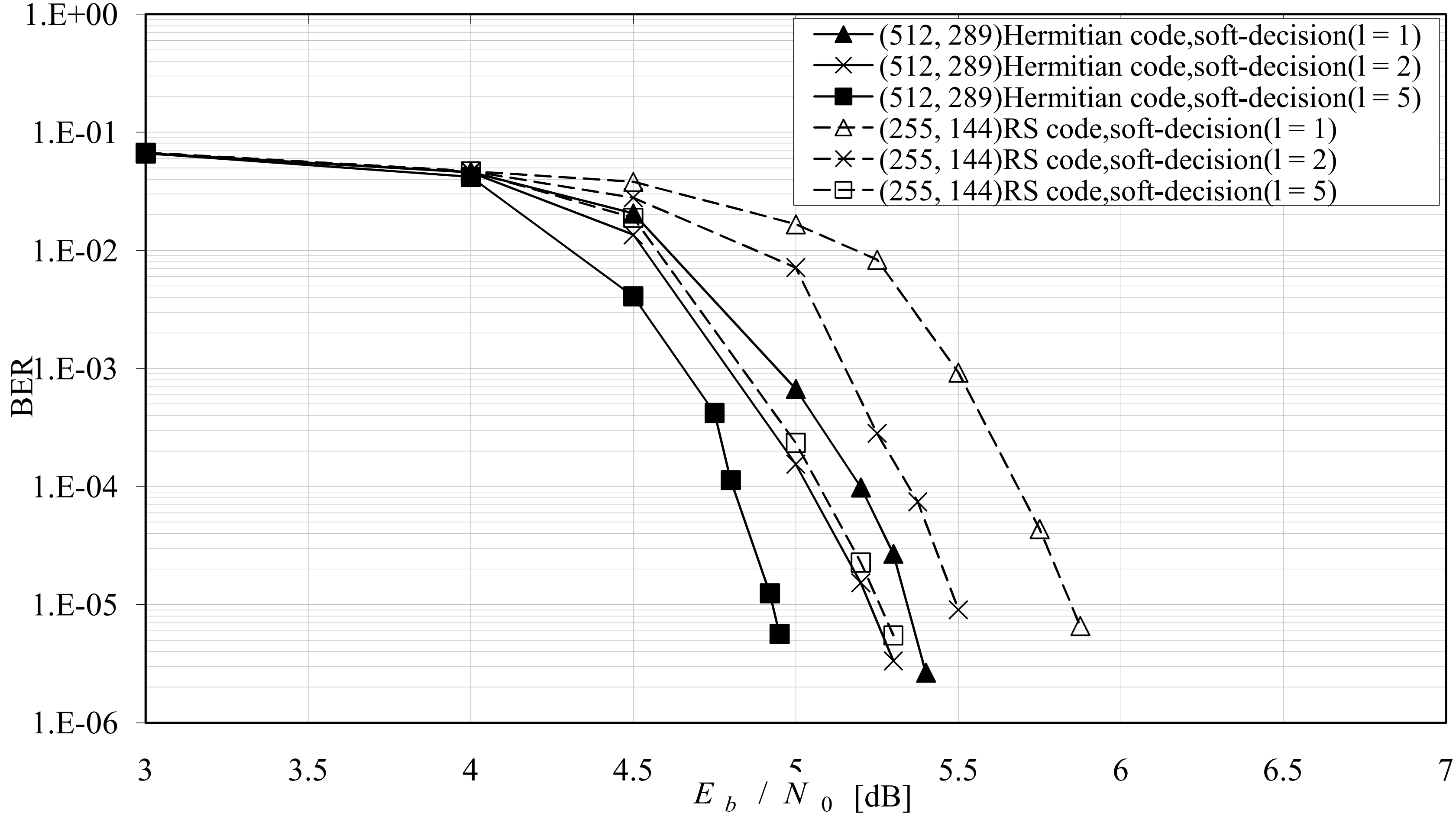
Both codes are defined in GF(64), over AWGN channel




| Output size \ Codes | Hermitian (512, 289) | RS (63, 35) | RS (255, 144) |
|---|---|---|---|
| $l = 1$ | $C = 892$ | $C = 103$ | $C = 430$ |
| $l = 2$ | $C = 1813$ | $C = 204$ | $C = 859$ |
| $l = 5$ | $C = 4602$ | $C = 715$ | $C = 3004$ |

# Ⅰ.Hermitian code ~ RS code

Hermitian code is defined in GF(64) and RS code is defined in GF(256)



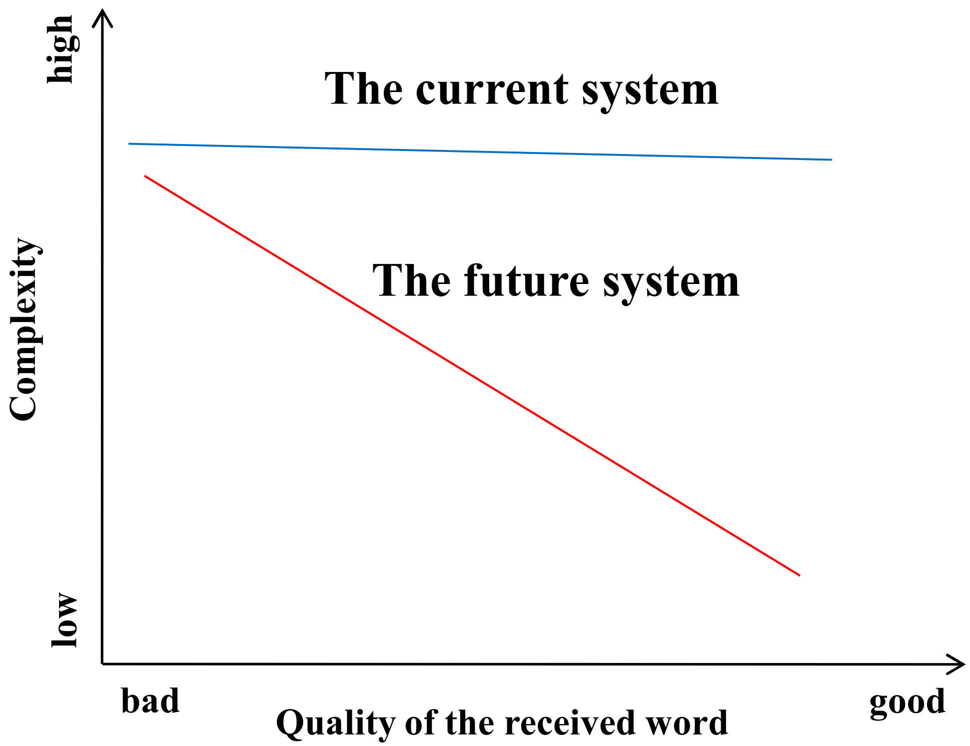

| Codes / Output size | Hermitian (512, 289) | RS (63, 35) | RS (255, 144) |
|---|---|---|---|
| $l = 1$ | $C = 892$ | $C = 103$ | $C = 430$ |
| $l = 2$ | $C = 1813$ | $C = 204$ | $C = 859$ |
| $l = 5$ | $C = 4602$ | $C = 715$ | $C = 3004$ |

# II. Modernised algebraic decoding

- Challenges → Inspirations
- Modernisation: Progressive algebraic soft decoding (PASD)
- Complexity reduction and performance evaluation

# II. Challenges → Inspirations

- The algebraic soft decoding is of high complexity, mainly due to the iterative interpolation process
- A rebound thinking – a common phenomenon for most of the modern decodings

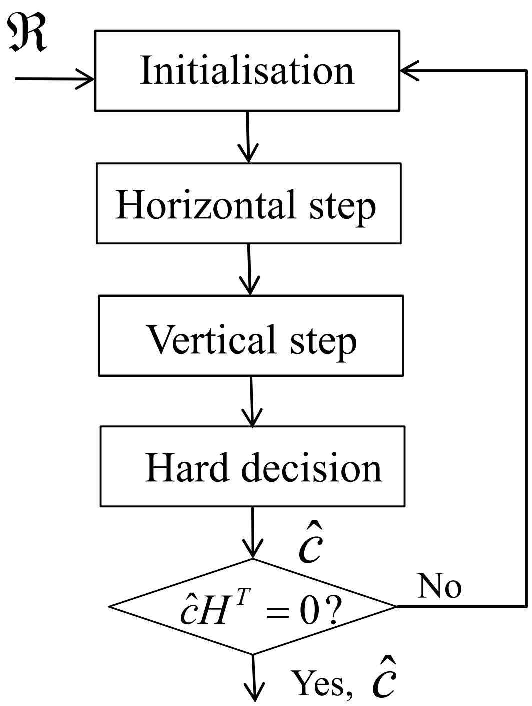



**Inspiration:** Can we design an algebraic decoder which can also adjust its complexity according to the quality of the received word?

We can 'borrow' the idea from iterative decoding!

# II. Challenges → Inspirations

- A review towards the modern codes (LDPC or Turbo codes)
  - The Belief Propagation (BP) algorithm with a parity check matrix H




- An iterative process
- Incremental computations between iterations
- A continue test of the decoding output
- Decoding capability and complexity can be adjusted according to the quality of $\Re$

# II. Modernised algebraic decoding

- The existing complexity reduction approaches

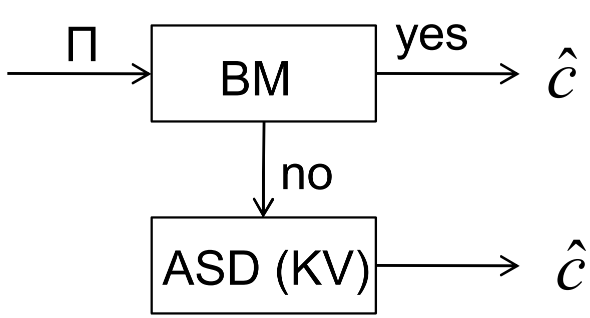
- Facilitated reliability transform: $M = \lfloor \lambda \cdot \Pi \rfloor$ **[Gross06]**

- Coordinate transform: $\{(\alpha_0, y_0), (\alpha_1, y_1), \ldots, (\alpha_{k-1}, y_{k-1}), (\alpha_k, y_k), \ldots, (\alpha_{n-1}, y_{n-1})\}$

  $\{(\alpha_0, 0), (\alpha_1, 0), \ldots, (\alpha_{k-1}, 0), (\alpha_k, y_k'), \ldots, (\alpha_{n-1}, y_{n-1}')\}$ **[KoetterITW03]**

- Elimination of unnecessary polynomials: $\boldsymbol{G} = \{Q \mid \text{lod}(Q) \leq C_M\}$ **[Chen07]**

- Hybrid decoding: **[Gross06]**



# II. Construction of a $(n, k)$ RS code

- The message polynomial evaluation
  - Let $\boldsymbol{u} = (u_0, u_1, \ldots, u_{k-1}) \in \mathrm{GF}(q)$ be a message vector, forming a message polynomial:

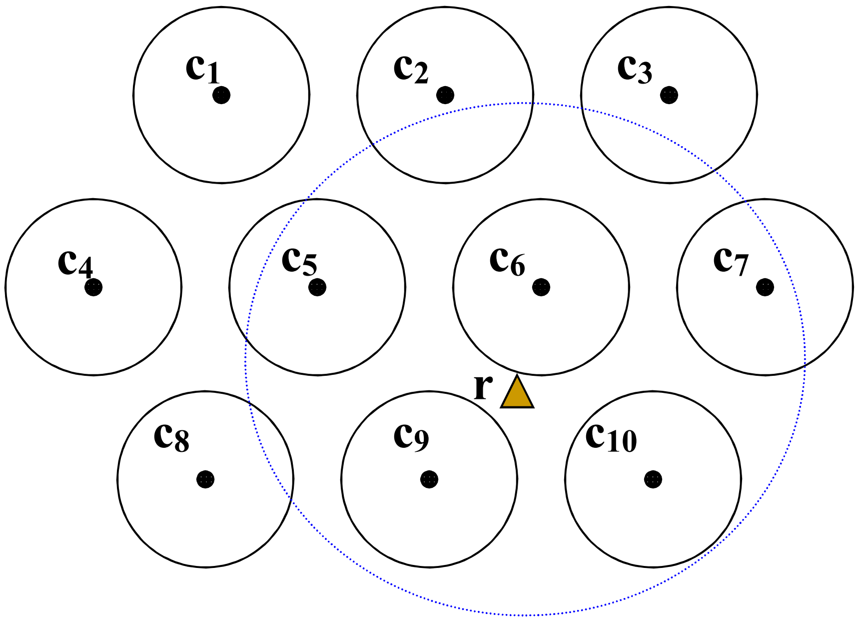
$$u(x) = u_0 + u_1 x + \cdots + u_{k-1} x^{k-1}$$

  - Choosing $n$ ($n \leq q$) distinct elements $\alpha_0, \alpha_1, \ldots, \alpha_{n-1} \in \mathrm{GF}(q)\backslash\{0\}$, the output codeword **c** can be generated as

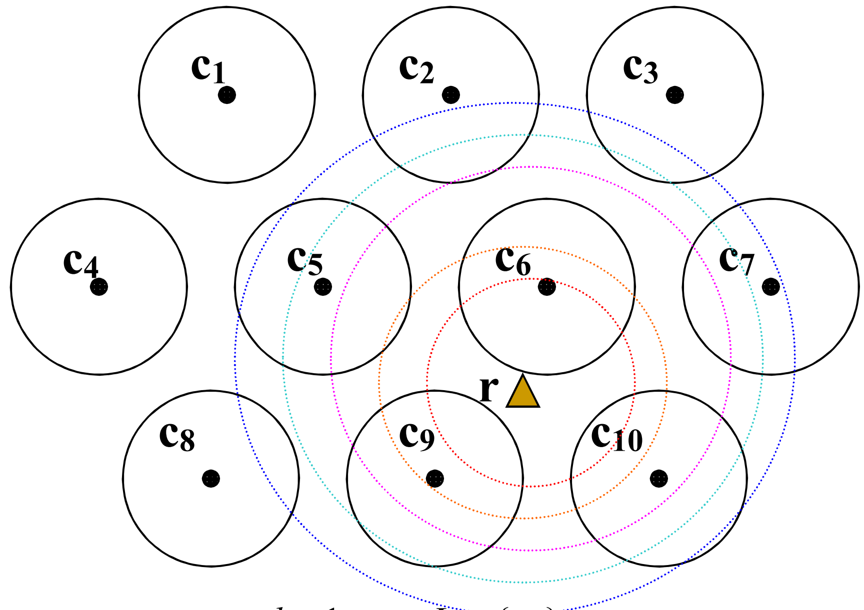
$$\mathbf{c} = (c_0, c_1, \ldots, c_{n-1}) = (u(\alpha_0), u(\alpha_1), \ldots, u(\alpha_{n-1}))$$

# II. A graphical thinking

**ASD** ( $l = 5$ ) ⟶ **PASD** ( $l = 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$ )




$$L = \{c_5, c_6, c_7, c_9, c_{10}\}$$




$l = 1 \rightarrow L = \{c_6\}$

$l = 2 \rightarrow L = \{c_6, c_9\}$

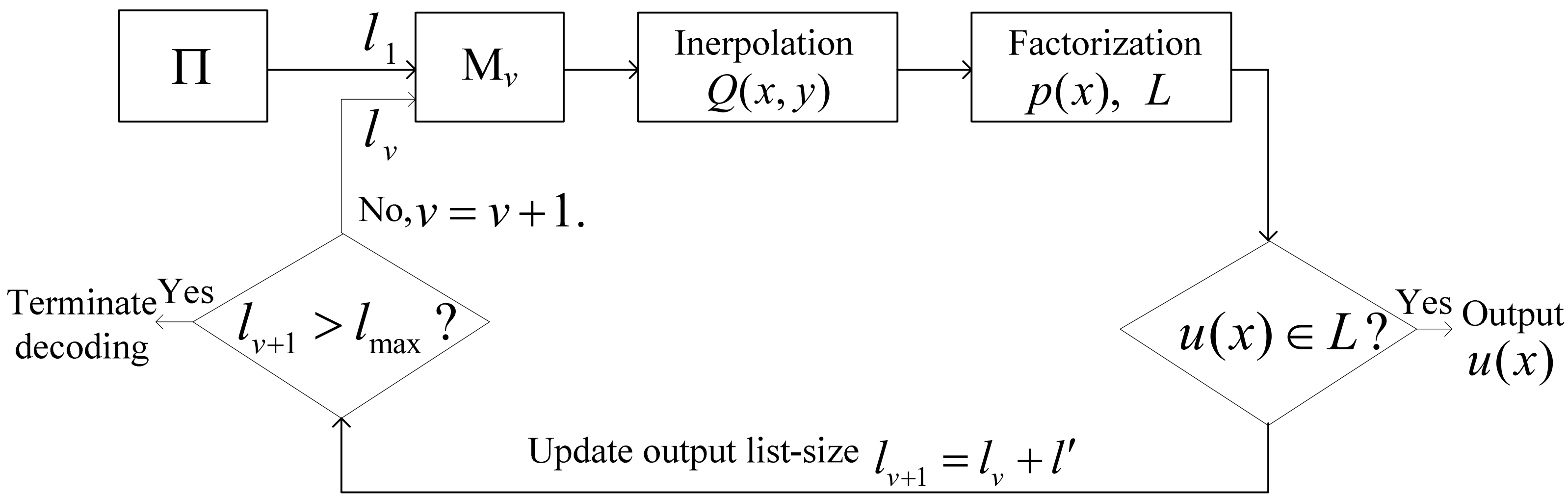
$l = 3 \rightarrow L = \{c_6, c_9, c_{10}\}$

$l = 4 \rightarrow L = \{c_5, c_6, c_9, c_{10}\}$

$l = 5 \rightarrow L = \{c_5, c_6, c_7, c_9, c_{10}\}$

If $\mathbf{c}_6$ is the transmitted codeword, PASD completes the decoding with $l = 1$ rather than $l = 5$ as the KV algorithm – Optimizing the assignment of decoding parameters & complexity.

# Ⅱ. The PASD decoding system




$l_v$- designed output list size at each iteration;
$l_{max}$- the designed maximal output list size;
$l'$- step size for updating the output list size;
$L$- the output list of all polynomials $p(x)$ such that $y$-$p(x)|Q(x, y)$.

Two key steps: Progressive Reliability Transform (PRT) → $M_1$, $M_2$, …, $M_v$, …
Progressive Interpolation (PIP) → $Q_1(x, y)$, $Q_2(x, y)$, …, $Q_v(x, y)$, …

**[Tang11]**

# II. Defining the zero condition constraints

- Multiplicity $m_{ij}$ ~ interpolated point $(x_j, \alpha_i)$
- Given a polynomial $Q(x, y)$, $m_{ij}$ implies $\boldsymbol{D}_{r,s}(Q(x, y))|_{x=xj,\ y=\alpha i} = 0$ for $r + s < m_{ij}$

- **Definition 1:** Let $\Lambda(m)$ denotes a set of zero condition constraints $(r, s)$ indicated by $m$, then $\Lambda(M)$ denotes a collection of all the sets $\Lambda(m_{ij})$ defined by the entry $m_{ij}$ of M

$$\Lambda(\mathrm{M}) = \{\Lambda(m_{ij}),\ m_{ij} \in \mathrm{M}\}$$

- Example:

$$M = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{aligned}\Lambda(\mathrm{M}) = \{ & \{(0,\ 0), (1, 0), (0, 1)\}_{00}, \varnothing_{01}, \varnothing_{02}, \\ & \varnothing_{10}, \{(0, 0)\}_{11}, \{(0, 0)\}_{12} \\ & \{(0, 0)\}_{20}, \{(0, 0), (1, 0), (0, 1)\}_{21}, \varnothing_{22} \\ & \varnothing_{30}, \varnothing_{31}, \{(0, 0)\}_{32}\}\end{aligned}$$

# II. Defining the zero condition constraints

- **Definition 2:** Let $m_{ij}^{v}$ and $m_{ij}^{v+1}$ denote the entries of matrix $M_v$ and $M_{v+1}$, the incremental zero condition constraints introduced between the matrices are defined as a collection of all the residual sets between $\Lambda(m_{ij}^{v+1})$ and $\Lambda(m_{ij}^{v})$ as:

$$\Lambda(\Delta M_{v+1}) = \Lambda(M_{v+1}) - \Lambda(M_v) = \{\Lambda(m_{ij}^{v+1}) - \Lambda(m_{ij}^{v})\}$$

- Example:

$$M_2 = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad M_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\Lambda(M_2) = \{\{(0,0),(1,0),(0,1)\}_{00}, \varnothing_{01}, \varnothing_{02}, \varnothing_{10}, \{(0,0)\}_{11}, \{(0,0)\}_{12} \{(0,0)\}_{20}, \{(0,0),(1,0),(0,1)\}_{21}, \varnothing_{22} \varnothing_{30}, \varnothing_{31}, \{(0,0)\}_{32}\}$$

10 constraints

$$\Lambda(M_1) = \{\{(0,0)\}_{00}, \varnothing_{01}, \varnothing_{02}, \varnothing_{10}, \varnothing_{11}, \{(0,0)\}_{12} \{(0,0)\}_{20}, \{(0,0)\}_{21}, \varnothing_{22} \varnothing_{30}, \varnothing_{31}, \{(0,0)\}_{32}\}$$

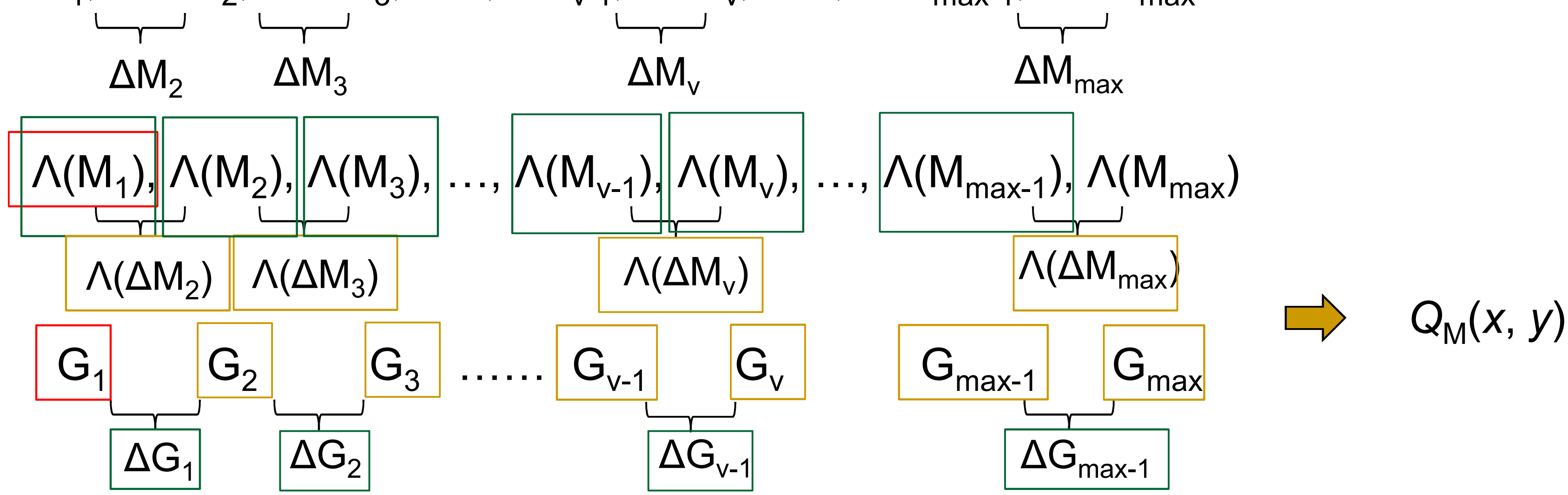
5 constraints

$$\Lambda(\Delta M_2) = \{\{(1,0),(0,1)\}_{00}, \varnothing_{01}, \varnothing_{02}, \varnothing_{10}, \{(0,0)\}_{11}, \varnothing_{12} \varnothing_{20}, \{(1,0),(0,1)\}_{21}, \varnothing_{22} \varnothing_{30}, \varnothing_{31}, \varnothing_{32}\}$$

5 constraints

# II. Progressive Interpolation

- PIP ($\Lambda$(M), G) – the Interpolation process that involves a group of polynomials G with respect to constraints of $\Lambda$(M).
- $M_1$, $M_2$, $M_3$, … , $M_{v-1}$, $M_v$, …, $M_{max-1}$, $M_{max}$



# II. Progressive interpolation

- PIP (Λ(M), G) – the Interpolation process that involves a group of polynomials G with respect to constraints of Λ(M).

- $$
\begin{aligned}
IP\,(\Lambda(M_{max}), G_{max}) \rightarrow\ & PIP(\Lambda(M_1), G_1)\, + \\
& PIP(\Lambda(M_1), \Delta G_1) + PIP(\Lambda(\Delta M_2), G_2)\, + \\
& PIP(\Lambda(M_2), \Delta G_2) + PIP(\Lambda(\Delta M_3), G_3)\, + \\
& \vdots \\
& PIP(\Lambda(M_{v-1}), \Delta G_{v-1}) + PIP(\Lambda(\Delta M_v), G_v)\, + \\
& \vdots \\
& PIP(\Lambda(M_{max-1}), \Delta G_{max-1}) + PIP(\Lambda(\Delta M_{max}), G_{max})
\end{aligned}
\longrightarrow \boxed{\text{Factorisation}}
$$

- The number of ‘factorisations’ has been increased. However, its complexity is rather marginal compared to interpolation.

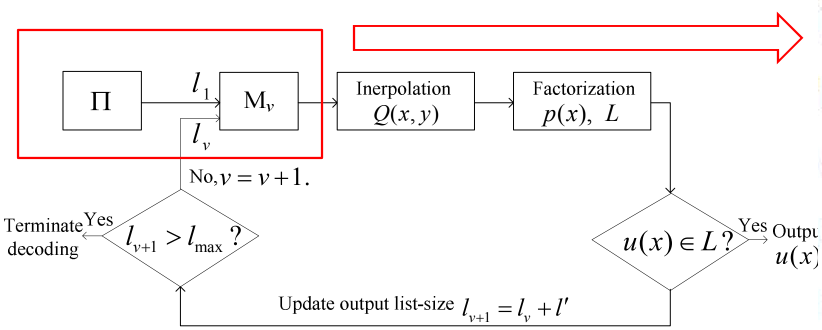
# II.Implementation algorithms

- Progressive Reliability Transform (*PRT*), producing

  $M_1, M_2, M_3, \ldots\ldots, M_v, \ldots\ldots, M_{max}$

- The output list size $l_v$ is determined by $l_v = \left\lfloor \frac{\Delta_{1,k-1}(C(M_v))}{k-1} \right\rfloor$

- $\Delta_{1,k-1}C(M_v) = \deg_{1,k-1}(x^a y^b \mid \mathrm{ord}(x^a y^b) = C(M_v))$




**Algorithm** Reliability transform with stopping criterion $l_v$

**Input:** Reliability matrix $\Pi$, $\Pi^*_{v-1}$, and the maximal output list size $l_v$ and multiplicity matrix $M_{v-1}$.

**Output:** Multiplicity matrix $M_v$.

**step 1:** Initiate $\Pi^*_v = \Pi^*_{v-1}$, $M_v = M_{v-1}$;

**step 2:** Find the largest entry $\pi^*_{ij}$ in $\Pi^*_v$ with the position $(i, j)$;

**step 3:** Update $\pi^*_{ij} = \frac{\pi_{ij}}{m_{ij}+2}$;

**step 4:** Update $m_{ij} = m_{ij} + 1$;

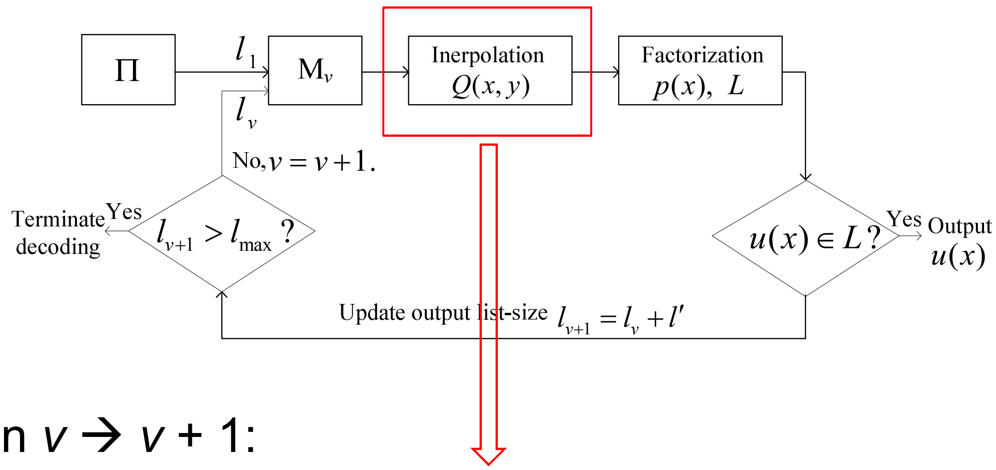
**step 5:** Compute $C(M_v) = \frac{1}{2}\sum_{i=0}^{q-1}\sum_{j=0}^{n-1} m_{ij}(m_{ij}+1)$;

**step 6:** Compute $l^*_v = \left\lfloor \frac{\Delta_{1,k-1}(C(M_v))}{k-1} \right\rfloor$;

**step 7:** If $l^*_v > l_v$, return $M_v$; otherwise go to step 2.

# II. Implementation algorithms

- Progressive Interpolation (*PIP*)




- From iteration $v$ → $v + 1$:

  1) Generate an incremental polynomial group

$$\Delta G_V = \{y^{l_v+1}, y^{l_v+2}, \ldots, y^{l_{v+1}}\}$$

Perform *PIP*(Λ($M_v$), $\Delta G_v$) → $\Delta G_v$', then update the new polynomial group as

$$G_{v+1} = G_v \cup \Delta G_v'$$

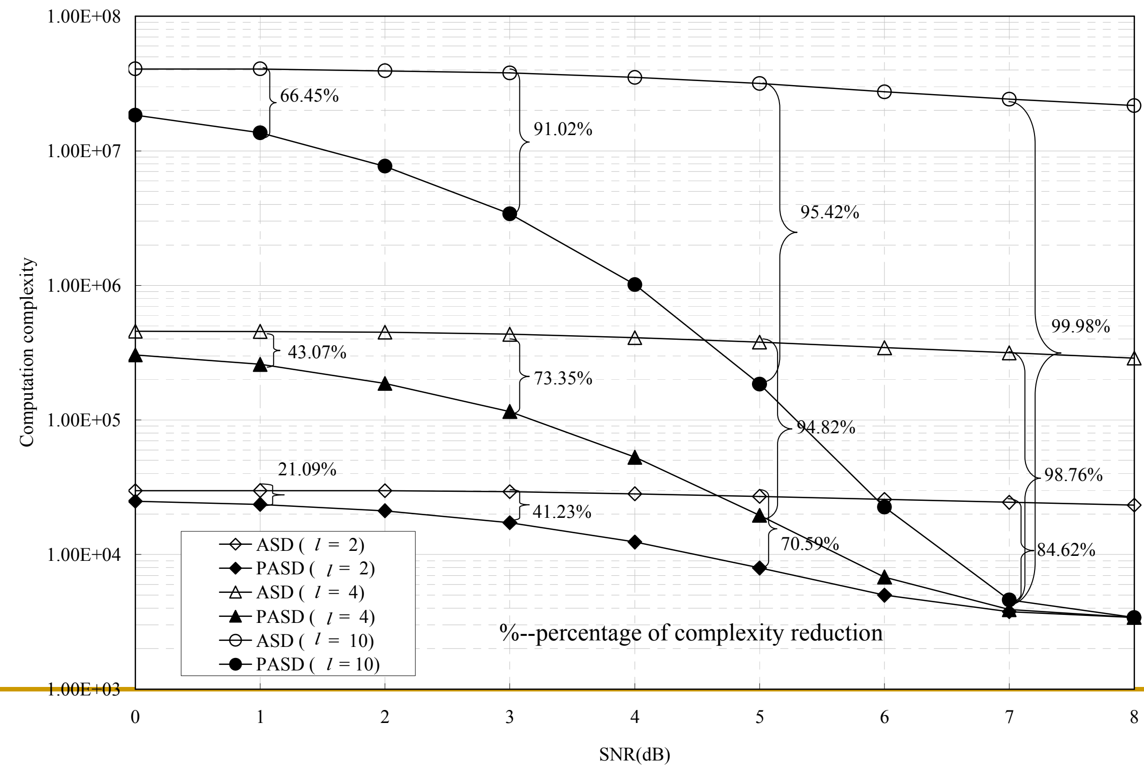
2) For the updated polynomial group $G_{v+1}$, perform *PIP* (Λ($\Delta M_{v+1}$), $G_{v+1}$) → $G_{v+1}$'.

# II. Complexity reduction

- Computational complexity (***O***): the averaged number of finite field arithmetic operations for decoding one codeword frame;
- Complexity reduction (Θ):

$$\Theta = \frac{O_{ASD} - O_{PASD}}{O_{ASD}} \times 100\%$$

- The (15, 5) RS code



# II. Complexity reduction

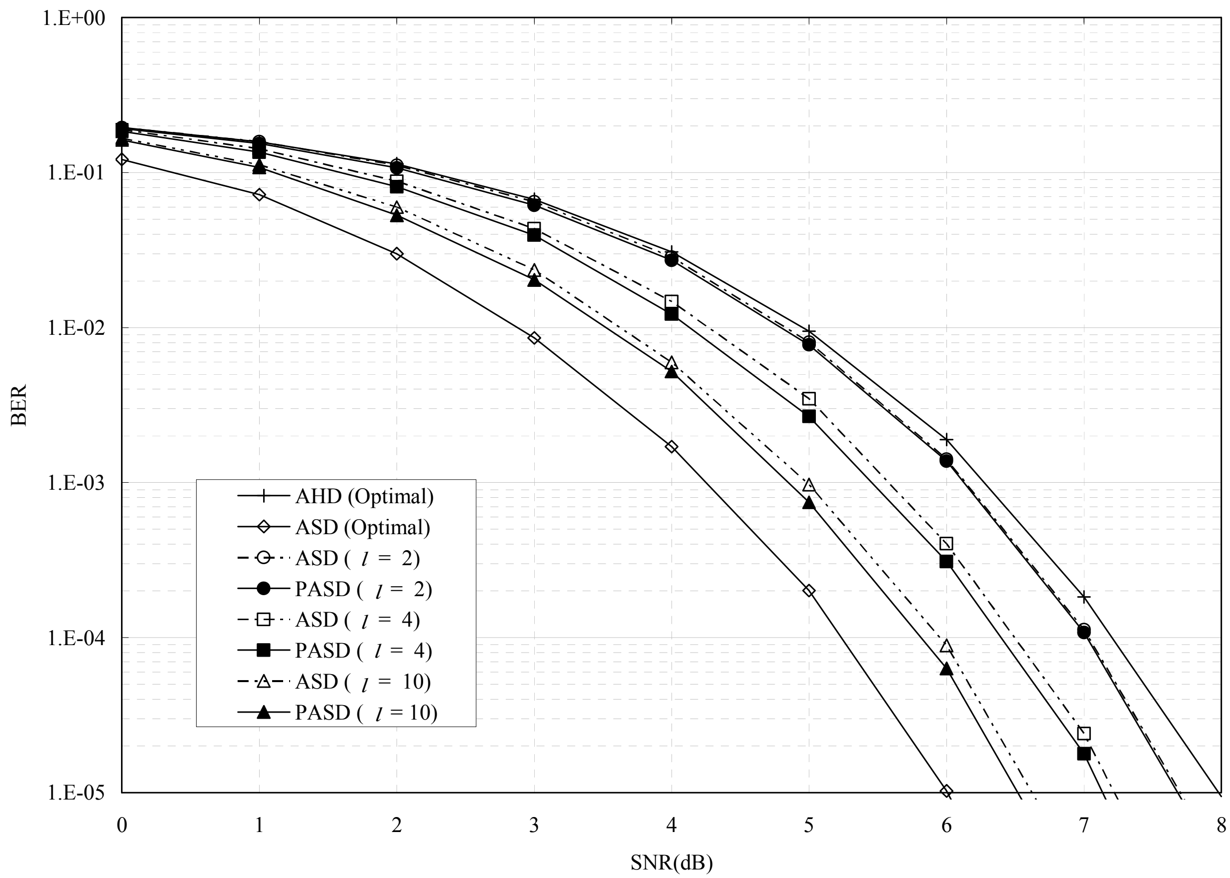
- Measurement of the decoding parameter *l*

Measure the assignment of $l$ with respect to the channel quality for (15,5) RS code

| $l$ / $SNR$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 2dB | 21.2130 | 15.8959 | 10.2188 | 7.0340 | 5.2340 | 4.0986 | 2.6862 | 2.6031 | 1.7170 | 29.2994 |
| 5dB | 81.0490 | 12.7920 | 3.2638 | 1.0861 | 0.5532 | 0.3028 | 0.1745 | 0.1230 | 0.1048 | 5.5078 |
| 8dB | 99.9339 | 0.0638 | 0.0014 | 0.0004 | 0.0003 | 0 | 0.0002 | 0 | 0 | 0 |

# II. Performance evaluatioin

- The (15, 5) RS code with BPSK, over AWGN channel



# II. Performance evaluation

- Successful decoding criterion: $S_M(\bar{c}) > \deg_{1,k-1}(Q(x, y))$
- Conventional ASD algorithm might ‘overkill’ the decoding problem
- Example: performing ASD and PASD with $l = 10$

| | KV(ASD) | | | PASD | | |
|---|---|---|---|---|---|---|
| $l$ | $S_M(C)$ | | $\deg_{1,k-1} Q(x,y)$ | $S_M(C)$ | | $\deg_{1,k-1} Q(x,y)$ |
| 1 | 4 | < | 8 | 4 | < | 8 |
| 2 | 10 | < | 12 | 10 | < | 12 |
| 3 | 13 | < | 16 | 13 | < | 16 |
| 4 | 19 | < | 20 | 19 | < | 20 |
| 5 | 21 | < | 24 | 21 | < | 24 |
| 6 | 27 | < | 28 | 27 | < | 28 |
| 7 | 30 | < | 32 | 30 | < | 32 |
| 8 | 34 | < | 36 | 34 | < | 36 |
| 9 | 41 | > | 40 | 41 | > | 40 |
| 10 | 44 | = | 44 | — | | — |

An example based on (15,5) RS code for understanding why the PASD algorithm can outperform the ASD algorithm

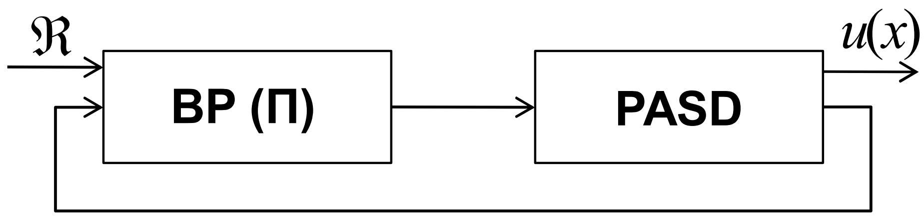
# Conclusions

- Construction of a Hermitian code and some of its properties;
- Hermitian code can be a promising candidate to replace RS code in future applications
- Algebraic soft-decoding of Hermitian codes, including the interpolated zero condition, validity of the decoding, optimal performance bound and complexity reduction approaches.
- Modernised algebraic soft decoding algorithm: a progressive approach
- Two key steps of PASD: progressive reliability transform & progressive interpolation
- Optimises both decoding complexity and performance
- A general approach for all sorts of algebraic decoding problems.

# Future work

- A continue thinking:
  PASD algorithm → performance ~ Π dependent;
  → complexity ~ Π dependent;

- An priori process to the PASD algorithm can be introduced to enhance the reliability of Π, enabling both a performance improvement and a faster convergence of decoding complexity.



# References


- [Guruswami99] V. Guruswami and M. Sudan, “Improved decoding of Reed-Solomon and algebraic-geometric codes,” *IEEE Trans. Inform. Theory,* vol. 45, pp.1757-1767, 1999.
- [Koetter03] R. Koetter and A. Vardy, “Algebraic soft-decision decoding of Reed-Solomon codes,” *IEEE Trans. Inform. Theory,* vol. 49, pp.2809-2825, 2003.
- [Chen09] L. Chen, R. Carrasco and M. Johnston, “Soft-decision list decoding of Hermitian codes,” *IEEE Trans. Commun.,* vol. 57, pp.2169-2176, 2009.
- [Lee10] K. Lee and M. O’Sullivan, “Algebraic soft-decision list decoding of Hermitian codes, ” *IEEE Trans. Inform. Theory,* vol. 56, pp.2587-2600, 2010.
- [Nielsen01] R. Nielsen, “List decoding of linear block codes,” PhD thesis, Lyngby, Demark Tech. Univ. Denmark, 2001.
- [Chen08] L. Chen, R. Carrasco and M. Johnston, “Reduced complexity interpolation for list decoding Hermitian codes,” *IEEE Trans. Wireless Commun.,* vol. 7, pp.4353-4361, 2008.
- [El-Khamy06] M. El-Khamy and R. McEliece, “Iterative algebraic soft-decision list decoding of Reed-Solomon codes,” *IEEE Journal on Selected Areas in Communications,* vol. 24, pp.481-489, 2006.
- [Chen07] L. Chen, R. Carrasco and E. Chester, “Performance of Reed-Solomon codes using the Guruswami-Sudan algorithm with improved interpolation efficiency,” *IEE Proc. Commun.,* vol. 1, pp.241-250, 2007.
- [Gross06] W. Gross, F. Kschischang, R. Koetter and P. Gulak, “Applications of algebraic soft-decision decoding of Reed-Solomon codes,” *IEEE Trans. Commun.,* vol. 54, pp.1224-1234, 2006.
- [KoetterITW03] R. Koetter and A. Vardy, “Complexity reducing transformation in algebraic list decoding of Reed-Solomon codes,” *Proc. IEEE Inform. Theory Workshop,* April, 2003.
- [Tang11] S. Tang, L. Chen and X. Ma, “Progressive list-enlarged algebraic soft decoding of Reed-Solomon codes,” *IEEE Commun. Lett.,* to be submitted, 2011.

# Acknowledgement

- The UK government Overseas Research Scholarship (ORS) scheme, supporting my PhD engagement (Part I of the presentation).

- The National Natural Science Foundation of China (NSFC), supporting the proposed work of Part II. Project: Advanced coding technology for future storage devices, ID: 61001094. Role: principle investigator (PI).

- Siyun Tang for implementing the PASD algorithm and Prof. Xiao Ma for his thoughtful discussion

Thank you!