Progressive Algebraic Soft-Decision Decoding of Reed-Solomon Codes

- Li Chen （陈立）, PhD, MIEEE
- Associate Professor, School of Information Science and Technology
- Sun Yat-sen University, China
- Joint work with Prof. Xiao Ma and Mrs. Siyun Tang

- Hangzhou, China
- 5th, Nov, 2011

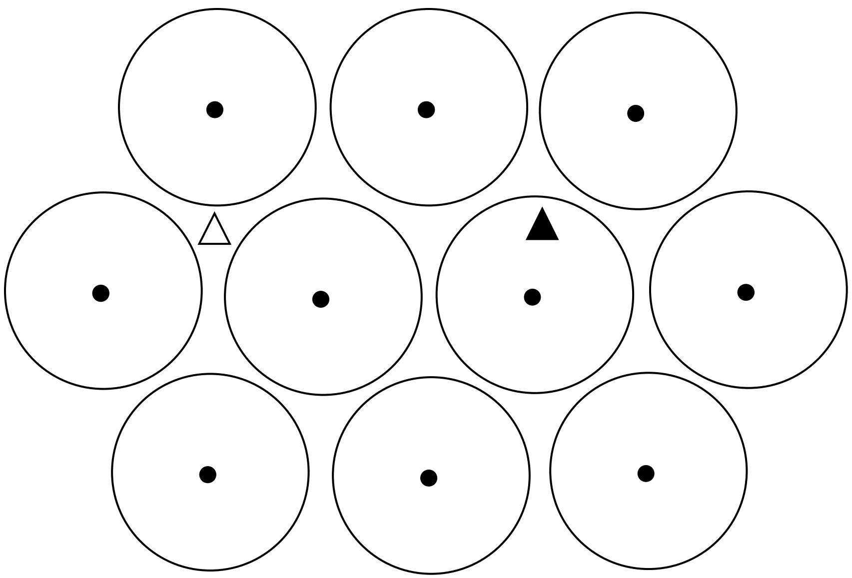
Outline

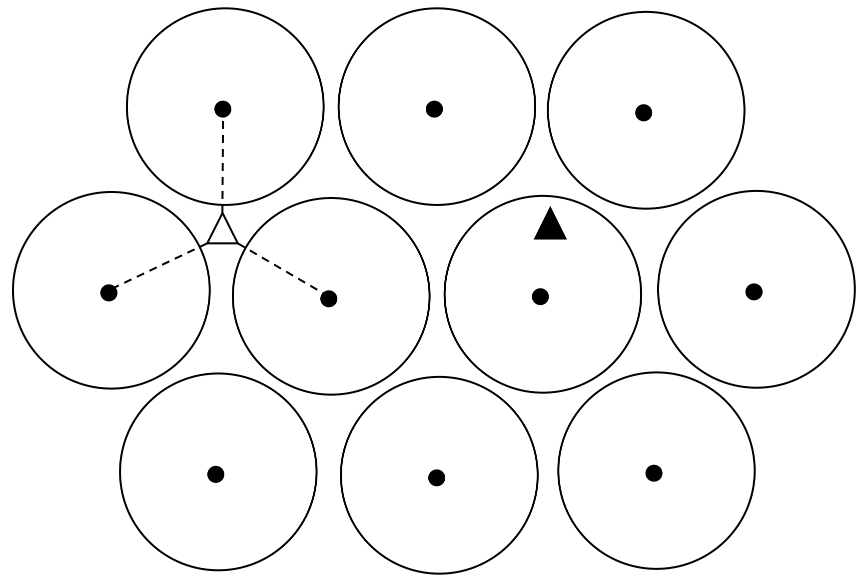
- Introduction
- Progressive ASD (PASD) algorithm
- Validity analysis
- Complexity analysis
- Error-correction performance
- Conclusions

I. Introduction of list decoding

- Decoding philosophy evolution of an (n, k) RS code

Unique decoding ⟶ **List decoding**





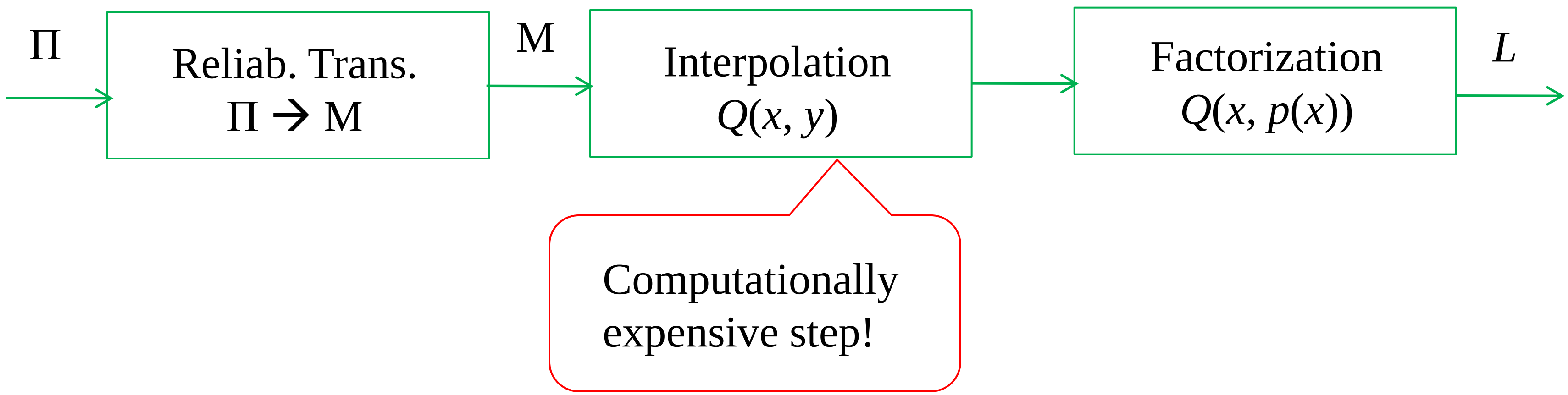
$$\tau_{unique} = \left\lfloor \frac{n-k}{2} \right\rfloor \longrightarrow \tau_{list} = n - \left\lfloor \sqrt{n(k-1)} \right\rfloor - 1$$

I. Overview of Enc. & Decd.

- Encoding
  - Given a message polynomial: $u(x) = u_0 + u_1 x + \cdots + u_{k-1}x^{k-1}$ $\quad (u_i \in \mathrm{GF}(q))$
  - Generate the codeword of an (n, k) RS code

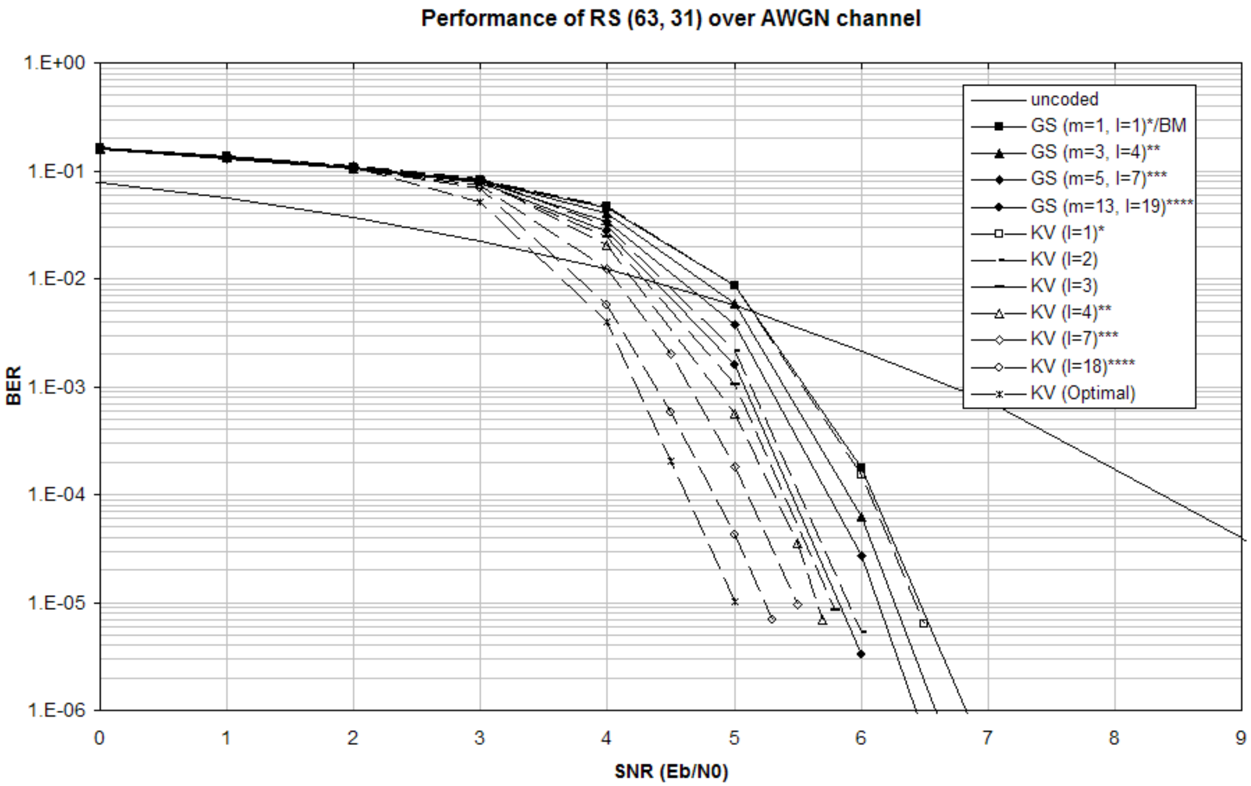
$$\mathbf{c} = (c_0, c_1, \ldots, c_{n-1}) = (u(\alpha_0), u(\alpha_1), \ldots, u(\alpha_{n-1})) \qquad (\alpha_i \in \mathrm{GF}(q)\backslash\{0\})$$

- Decoding



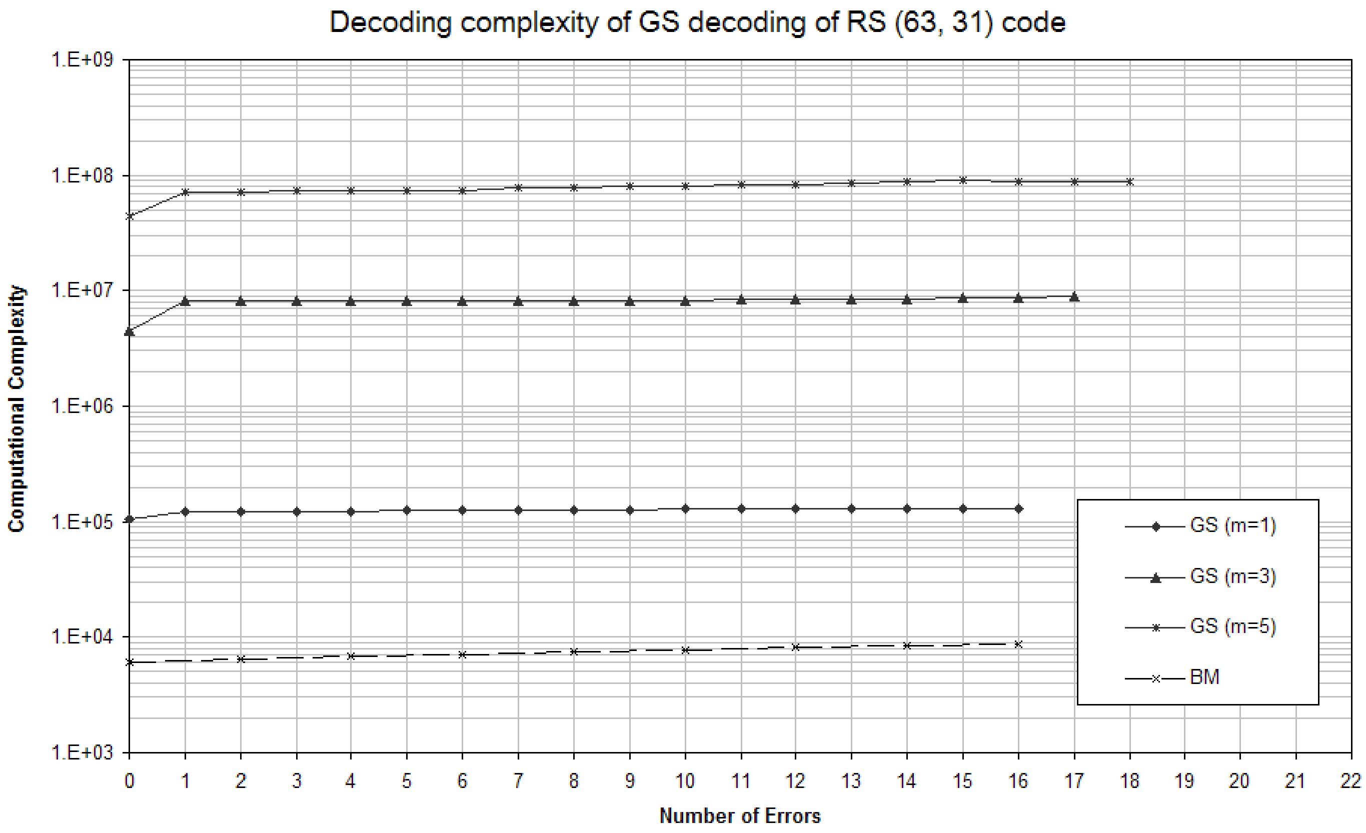
I. Perf. of AHD and ASD

- Algebraic hard-decision decoding (AHD) [Guruwammi99] (-GS Alg)
- Algebraic soft-decision decoding (ASD) [Koetter03] (-KV Alg)
- Advantage on error-correction performance



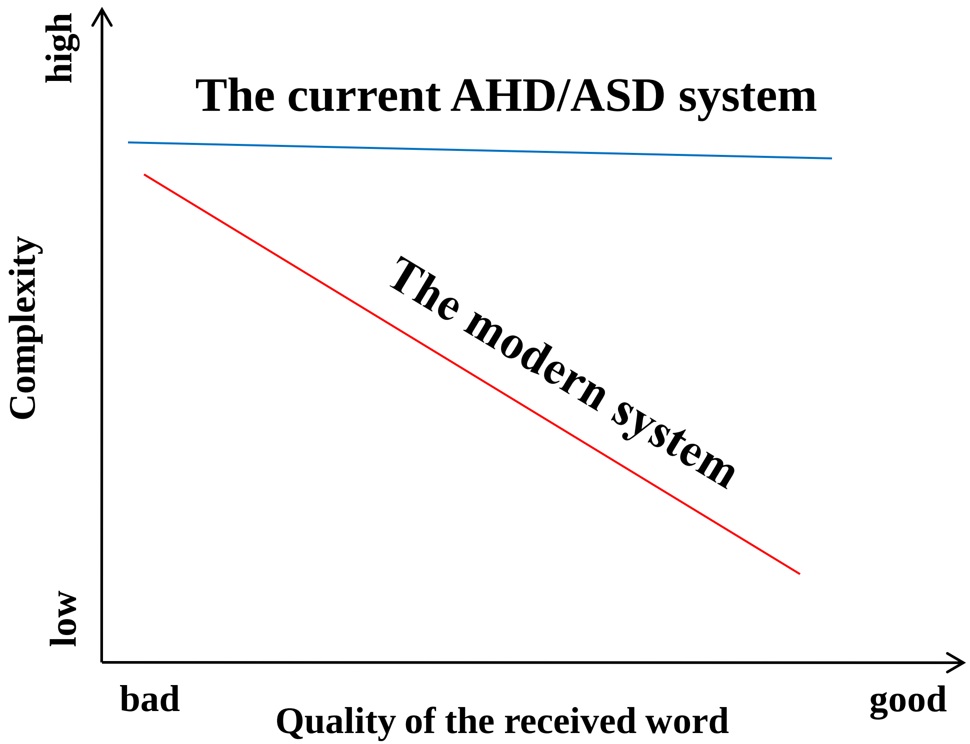
I. Introduction

- Price to pay: decoding complexity

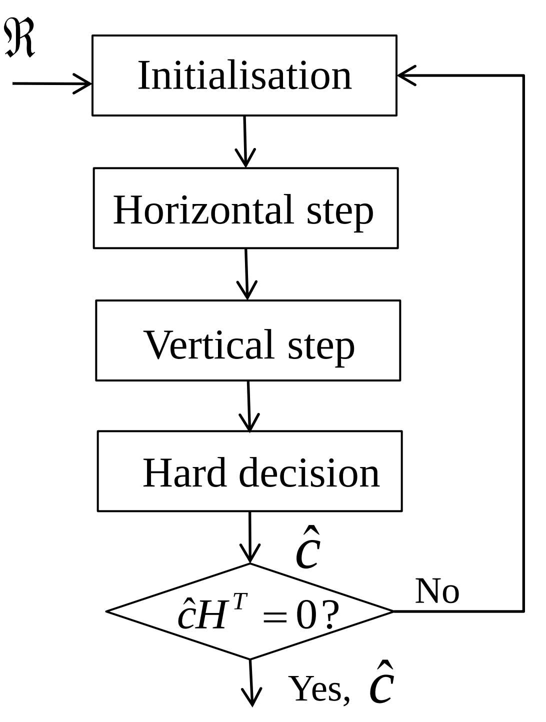


I. Inspirations

- The algebraic soft decoding is of high complexity. It is mainly due to the iterative *interpolation process*;
- A modernized thinking – the decoding should be *flexible* (i.e., channel dependent).



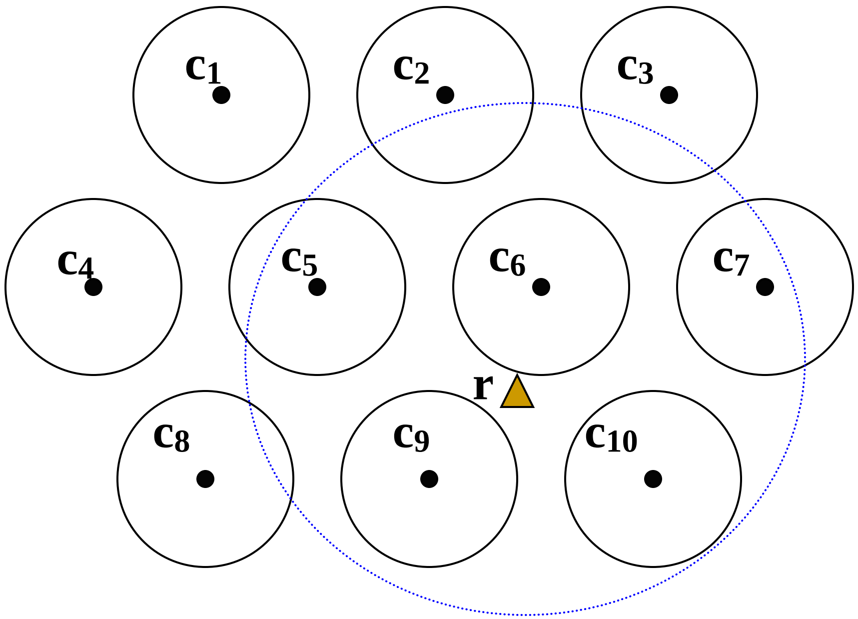
E.g., the belief propagation algorithm

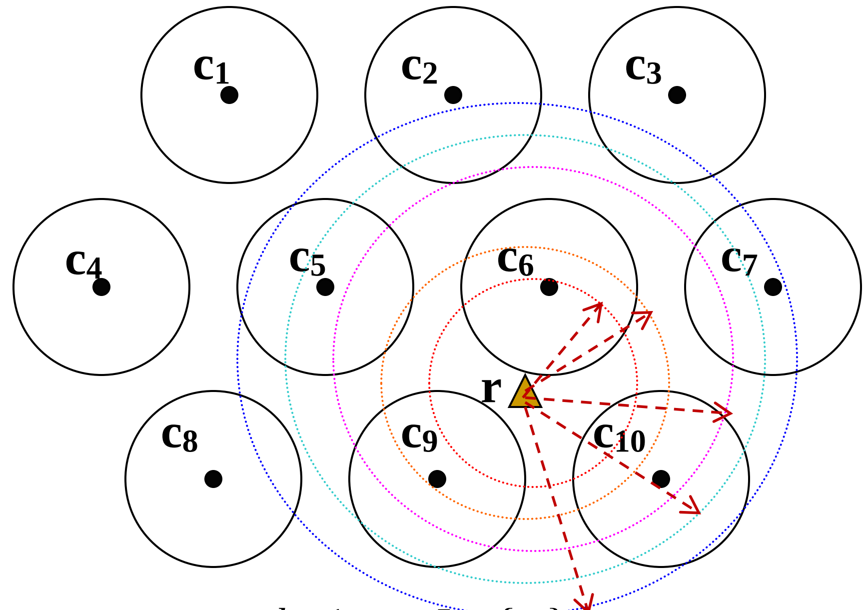


Iterative proc. + Incremental comp. + Continues validat.

II. A Graphical Introduction

ASD ($l=5$) $\longrightarrow$ **PASD** ($l=1\rightarrow2\rightarrow3\rightarrow4\rightarrow5$)





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

$|L|$ - factorization output list size

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

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

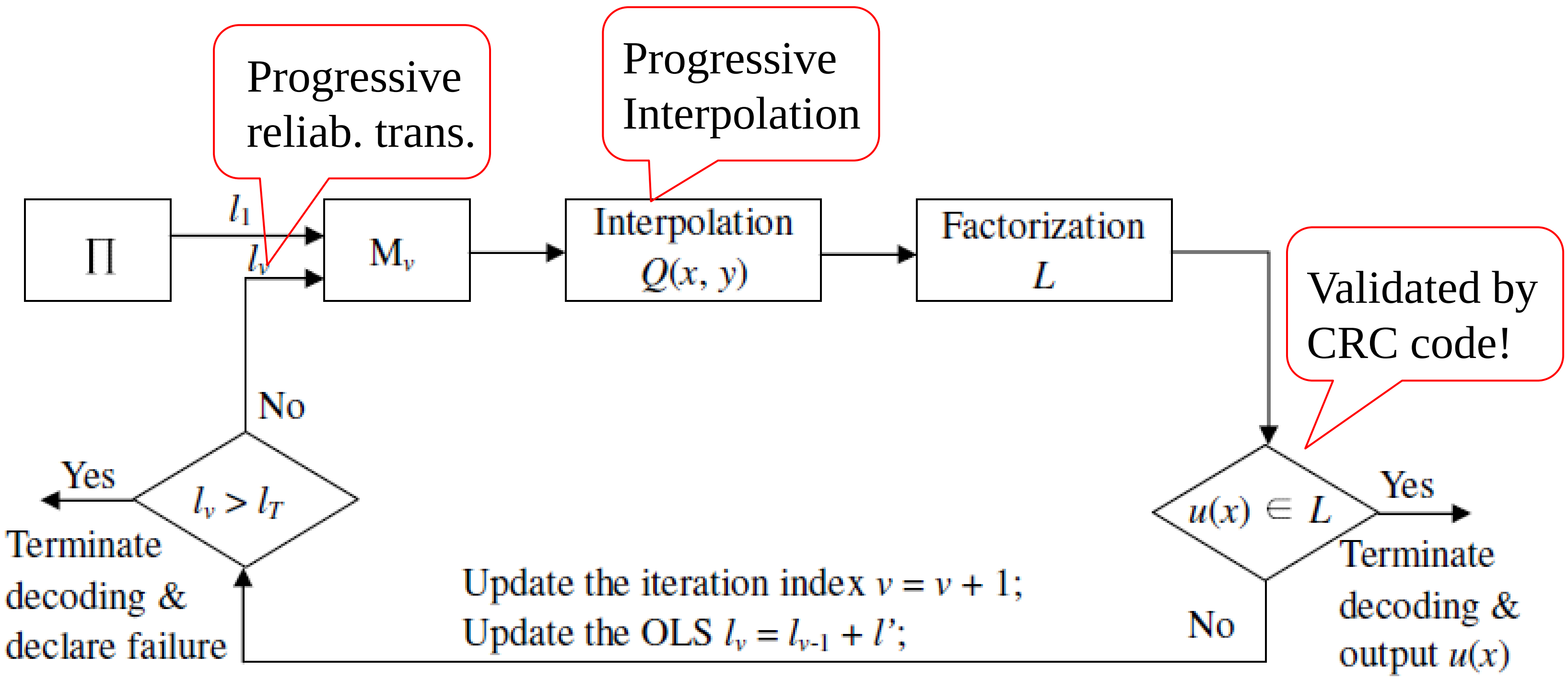
$l=3 \rightarrow L=\{c_6,c_9,c_{10}\}$

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

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

Enlarging the decoding radius progressively → Enlarging the factorization OLS progressively

Ⅱ. Decoding architecture



v – iteration index;

l_v -- designed OLS at each iteration;

l_T -- designed maximal OLS (~the maximal complexity that the system can tolerate);

l'– step size for updating the OLS, $l_{v+1} = l_v + l'$;

II. Progressive approach

- Enlarge the decoding radius → Enlarge the OLS

- Progressive decoding

Π

Series of OLS: l_1, l_2, ……, $l_{v\text{-}1}$, l_v, ……, l_T

Series of M mtxs.: M_1, M_2, ……, $M_{v\text{-}1}$, M_v, ……, M_T

Series of Q polys.: $Q^{(1)}$, $Q^{(2)}$, ……, $Q^{(v\text{-}1)}$, $Q^{(v)}$, ……, $Q^{(T)}$

$$l_v = \left\lfloor \frac{\Delta_{1,k-1}(\mathcal{C}(\mathrm{M}_v))}{k-1} \right\rfloor$$

Question: Can the solution of $Q^{(v)}$ be found based on the knowledge of $Q^{(v\text{-}1)}$?

II. Incremental interpolation constraints

- Multiplicity m_{ij} ~ interpolated point (x_j, α_i)
- Given a polynomial $Q(x, y)$, m_{ij} implies $\frac{1}{2}m_{ij}(m_{ij}+1)$ constraints of

$$\mathcal{D}_{r,s}(Q(x,y))|_{x=x_j,y=\alpha_i} = \sum_{a\geq r, b\geq s} \binom{a}{r}\binom{b}{s} Q_{ab} x_j^{a-r} \alpha_i^{b-s} = 0 \qquad (r + s < m_{ij})$$

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

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

- Example: $\mathrm{M} = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \Longrightarrow$ $\Lambda(\mathrm{M}) = \{(0, 0)_{00}, (1, 0)_{00}, (0, 1)_{00}, (0, 0)_{11}, (0, 0)_{12}, (0, 0)_{20}, (0, 0)_{21}, (1, 0)_{21}, (0, 1)_{21}, (0, 0)_{32}\}$

II. Incremental Interp. Constr.

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

$$\Lambda(\Delta \mathrm{M}_v) = \{\Lambda(\mathrm{M}_v) \setminus \Lambda(\mathrm{M}_{v-1})\} = \{\Lambda(m_{ij}^{v}) \setminus \Lambda(m_{ij}^{v-1}),\ \forall\ m_{ij}^{v} \in \mathrm{M}_v \text{ and } m_{ij}^{v-1} \in \mathrm{M}_{v-1}\}$$

- Example: $\mathrm{M}_2 = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ $\quad \mathrm{M}_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

$\Lambda(\mathrm{M}_2) = \boxed{\{(0,0)_{00}, (1,0)_{00}, (0,1)_{00}, (0,0)_{11}, (0,0)_{12}, (0,0)_{20}, (0,0)_{21}, (1,0)_{21}, (0,1)_{21}, (0,0)_{32}\}}$

10 constraints

$\Lambda(\mathrm{M}_1) = \boxed{\{(0,0)_{00}, (0,0)_{12}, (0,0)_{20}, (0,0)_{21}, (0,0)_{32}\}}$

5 constraints

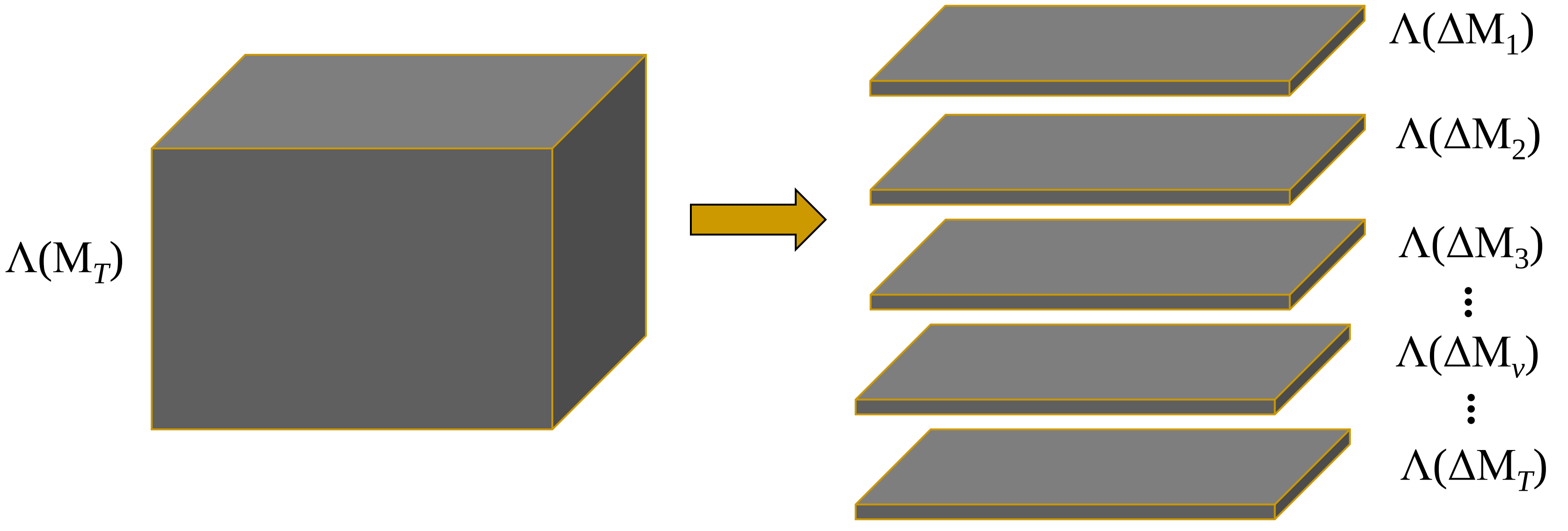
—

$\Lambda(\Delta\mathrm{M}_2) = \boxed{\{(1,0)_{00}, (0,1)_{00}, (0,0)_{11}, (1,0)_{21}, (0,1)_{21}\}}$

5 constraints

II. Progressive Interpolation

- A big chunk of interpolation task $\Lambda(\mathrm{M}_T)$ can be sliced into smaller pieces.



- $\Lambda(\mathrm{M}_T) = \Lambda(\Delta\mathrm{M}_1) \cup \Lambda(\Delta\mathrm{M}_2) \cup \cdots \cup \Lambda(\Delta\mathrm{M}_v) \cup \cdots \cup \Lambda(\Delta\mathrm{M}_T).$

- $\Lambda(\Delta\mathrm{M}_v)$ defines the interpolation task of iteration *v*.

II. Incremental Computations

- ***Review*** on the interpolation process – iterative polynomial construction
- Given M_v, the interpolation constraints are $\Lambda(\mathrm{M}_v)$
- The polynomial group is: $\mathbf{G}_v = \{g_0, g_1, \ldots, g_v\}$, $(\deg_y g_v = l_v)$

- The iterative process

$f_{(r,s)_{ij}}$ defines the outcome of the updated group.

For $(r, s)_{ij} \in \Lambda(\mathrm{M}_v)$

$$f_{(r,s)_{ij}} = \min\{g_t \mid \mathcal{D}_{(r,s)_{ij}}(g_t) \neq 0\}.$$

For each $g_t \in \mathbf{G}v$

$$g_t = \begin{cases} g_t, & \text{if } \mathcal{D}_{(r,s)_{ij}}(g_t) = 0 \\ [g_t, f_{(r,s)_{ij}}]_D, & \text{if } \mathcal{D}_{(r,s)_{ij}}(g_t) \neq 0 \text{ and } g_t \neq f_{(r,s)_{ij}} \\ [x f_{(r,s)_{ij}}, f_{(r,s)_{ij}}]_D, & \text{if } f_{(r,s)_{ij}}, \end{cases}$$

Finally, $Q^{(v)} = \min\{g_t \mid g_t \in \mathbf{G}_v\}$

II. Incremental Computations

- From iteration *v*-1 to *v* …
- The progressive interpolation can be seen as a ***progressive polynomial group expansion*** which consists of two successive stages.
- Let $\widetilde{\mathbf{G}}_{v-1} = \{\tilde{g}_0, \tilde{g}_1, \ldots, \tilde{g}_{l_{v-1}}\}$ be the outcome of iteration *v*-1.
- During the generation of $\widetilde{\mathbf{G}}_{v-1}$, a series of $f_{(r,s)_{ij}}$ with $(r, s)_{ij} \in \Lambda(M_{v-1})$ are identified and stored.

- ***Expansion I:*** expand the ***number*** of polynomials of the group

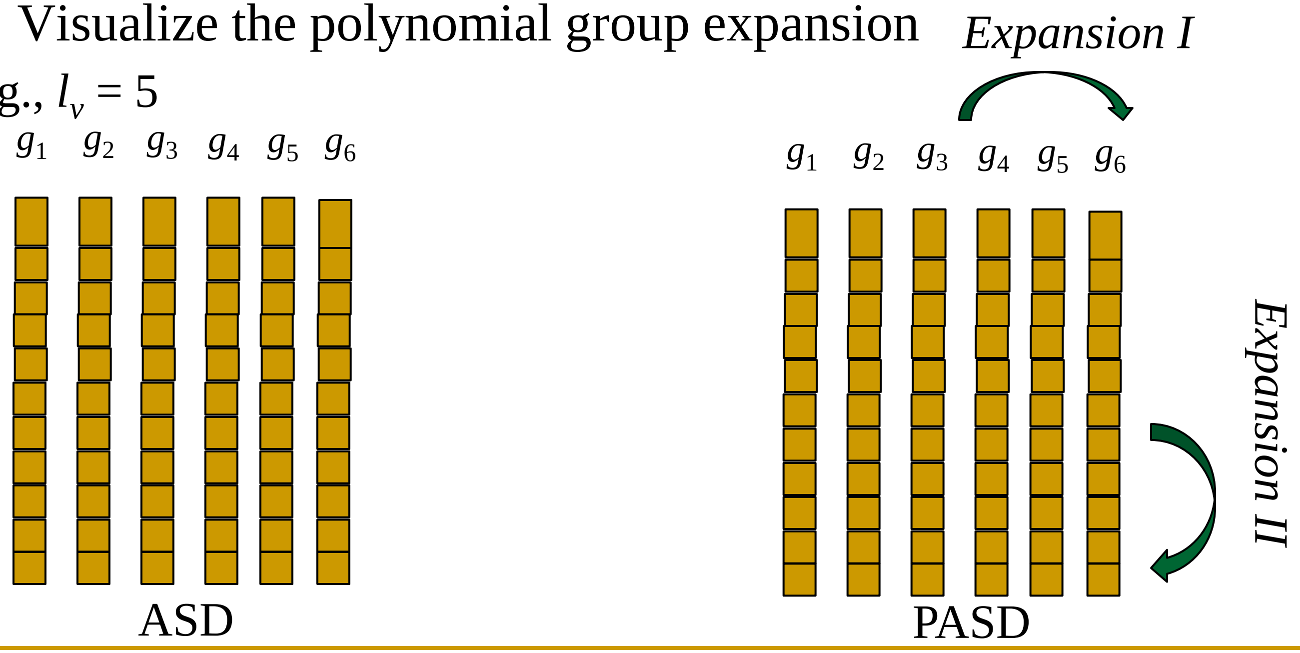
$$\Delta\mathbf{G}_v = \{g_{l_{v-1}+1}, \ldots, g_{l_v}\} = \{y^{l_{v-1}+1}, \ldots, y^{l_v}\}$$

- Polynomials of $\Delta\mathbf{G}_v$ perform interpolation w.r.t. constraints of $\Lambda(M_{v-1})$;
- Polynomials $f_{(r,s)_{ij}}$ are re-used for the update of $\Delta\mathbf{G}_v$.
- Let $\Delta\widetilde{\mathbf{G}}_v$ be the updated outcome of $\Delta\mathbf{G}_v$ and $\mathbf{G}_v = \widetilde{\mathbf{G}}_{v-1} \cup \Delta\widetilde{\mathbf{G}}_v$.

II. Incremental Computations

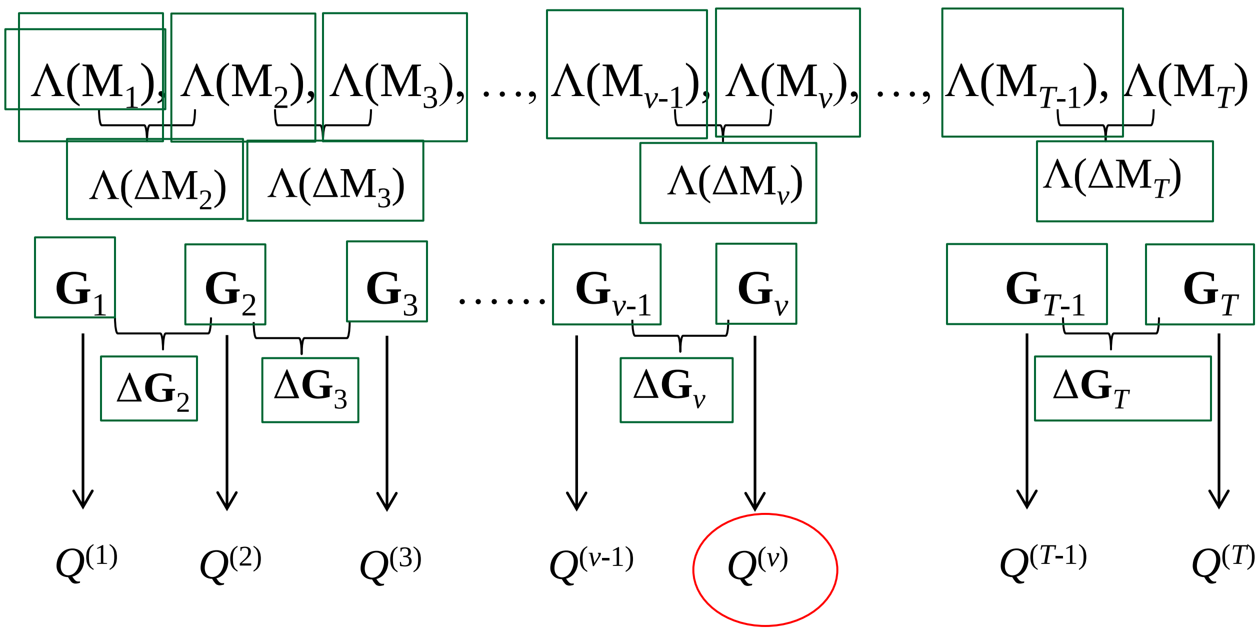
- ***Expansion II:*** expand the ***size*** of polynomials of the group $\mathbf{G}_v$
- Polynomials of $\mathbf{G}_v$ will now perform interpolation w.r.t. the incremental constraints $\Lambda(\Delta M_v)$, yielding $\tilde{\mathbf{G}}_v$.
- Finally, $Q^{(v)}(x, y) = \min\{\tilde{g}_t \mid \tilde{g}_t \in \tilde{\mathbf{G}}_v\}$
- ➔ Visualize the polynomial group expansion

E.g., $l_v = 5$



II. Progressive Interpolation

- The process of progressive interpolation
- $M_1, \quad M_2, \quad M_3, \ \ldots, \ M_{v-1}, \quad M_v, \ \ldots, \ M_{T-1}, \quad M_T$



If $Q^{(v)}(x, u(x)) = 0$, the decoding will be terminated.

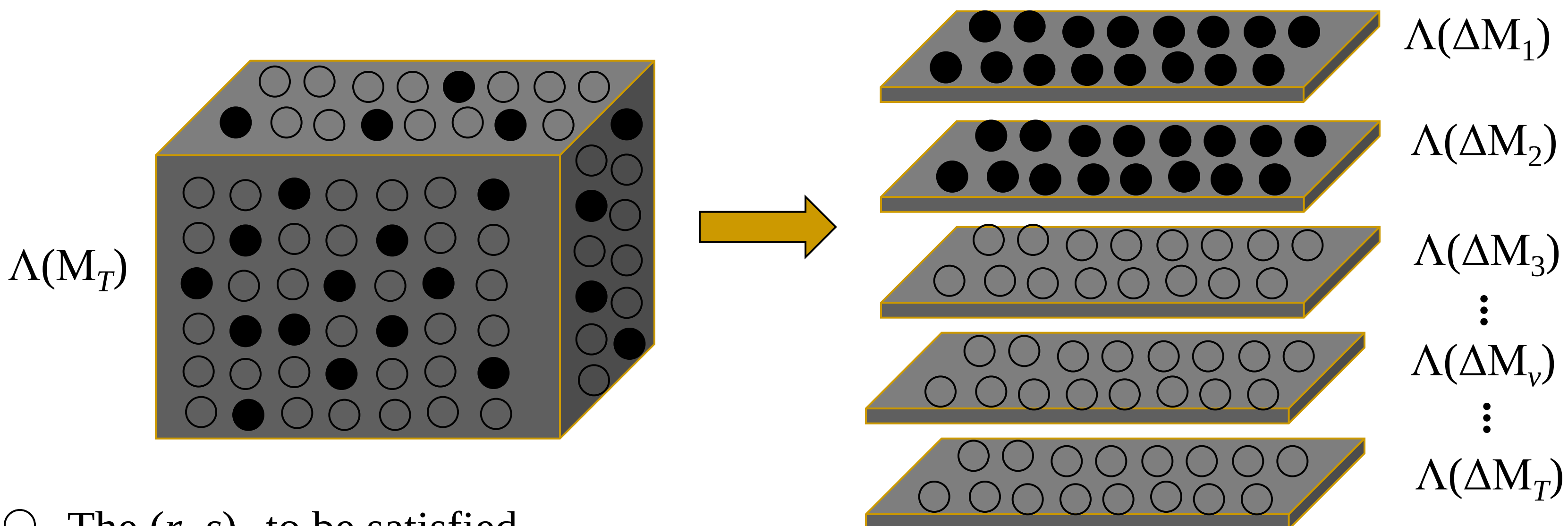
- Multiple factorizations are carried out in order to determine whether $u(x)$ has been found!

III. Validity Analysis

- For any two $(r_1, s_1)_{i1j1}$ and $(r_2, s_2)_{i2j2}$ of $\Lambda(\mathrm{M}_T)$:

  $(r_1, s_1)_{i1j1} \Longrightarrow (r_2, s_2)_{i2j2}$ <==> $(r_2, s_2)_{i2j2} \Longrightarrow (r_1, s_1)_{i1j1}$

- The algorithm imposes a *progressive interpolation order*



○ The $(r, s)_{ij}$ to be satisfied.

● The satisfied $(r, s)_{ij}$.

III. Validity Analysis

- Decoding with an OLS of l_v, the solution $Q(x, y)$ is seen as the minimal candidate chosen from the cumulative kernel $\overline{\mathcal{K}_{\mathcal{C}(\mathrm{M}_v)}}$

$$\overline{\mathcal{K}_{\mathcal{C}(\mathrm{M}_v)}} = \{Q \in \mathbb{F}_q[x, y] \mid \mathcal{D}_w(Q) = 0 \ \forall\ w \in \Lambda(\mathrm{M}_v),\ \ \deg_y Q \leq l_v\}$$

$$Q(x, y) = \min\{Q \in \overline{\mathcal{K}_{\mathcal{C}(\mathrm{M}_v)}}\}$$

- For both of the algorithms:

$$\Lambda(\mathrm{M}_v) = \Lambda(\Delta\mathrm{M}_1) \cup \Lambda(\Delta\mathrm{M}_2) \cup \cdots \cup \Lambda(\Delta\mathrm{M}_v)$$

the same set of constraints are defined for the cumulative kernel;

- Consequently, they will offer the same solution of $Q(x, y)$.

III. Validity Analysis

- In the end of *Expansion I*, $\mathbf{G}_v = \tilde{\mathbf{G}}_{v-1} \cup \Delta\tilde{\mathbf{G}}_v$
- Can $\tilde{\mathbf{G}}_{v-1}$ and $\Delta\tilde{\mathbf{G}}_v$ be found separately?
- Recall the polynomial updating rules

$$g_t = \begin{cases} g_t, & \text{if } \mathcal{D}_{(r,s)_{ij}}(g_t) = 0 \\ [g_t, f_{(r,s)_{ij}}]_D, & \text{if } \mathcal{D}_{(r,s)_{ij}}(g_t) \neq 0 \text{ and } g_t \neq f_{(r,s)_{ij}} \\ [x f_{(r,s)_{ij}}, f_{(r,s)_{ij}}]_D, & \text{if } f_{(r,s)_{ij}}, \end{cases}$$

• The minimal polynomial $f_{(r,s)_{ij}}$ defines the solution of one round of poly. update w.r.t. $(r, s)_{ij}$.
• If such a group expansion procedure does not change the identity of $f_{(r,s)_{ij}}$, $\tilde{\mathbf{G}}_{v-1}$ and $\Delta\tilde{\mathbf{G}}_v$ can indeed be found separately.

III. Validity Analysis

- *Lemma 1:* For all the polynomials $f_{(r,s)_{ij}}$ with $(r,s)_{ij} \in \Lambda(\mathrm{M}_{v-1})$, we have

$$\mathrm{lod}(f_{(r,s)_{ij}}) < \mathrm{lod}(y^{l_{v-1}+1}). \Longrightarrow (\, f_{(r,s)_{ij}} < y^{l_{v-1}+1}\,)$$

- Expansion I: update $\Delta\mathbf{G}_v$ to $\Delta\widetilde{\mathbf{G}}_v$ w.r.t. constraint of $\Lambda(\mathrm{M}_{v-1})$

For $(r,s)_{ij} \in \Lambda(\mathrm{M}_{v-1})$:

- No update is required
- Update is required
  - $f_{(r,s)_{ij}}$ is in memory, Since $f_{(r,s)_{ij}} < g^*$(of $\Delta\mathbf{G}_v$) $f_{(r,s)_{ij}}$ will be re-used
  - $f_{(r,s)_{ij}}$ is not in memory, it will be picked up from $\Delta\mathbf{G}_v$.

- The identity of the existing $f_{(r,s)_{ij}}$ is left unchanged. Consequently, the solution of $\widetilde{\mathbf{G}}_{v-1}$ remains intact.

- Therefore, $\mathbf{G}_v = \widetilde{\mathbf{G}}_{v-1} \cup \Delta\widetilde{\mathbf{G}}_v$

IV. Complexity Analysis

- Average decoding complexity – average number of finite field arithmetic operation for decoding one codeword frame;

- $\mathcal{P}_{l_v}$ --- the probability of the decoder is performing a successful decoding with an OLS of l_v;
  $\mathcal{O}_{l_v}$ --- the decoding complexity with an OLS of l_v;

- The average decoding complexity is:

$$\mathcal{O}_{\text{PASD}} = \underbrace{\sum_{v=1}^{T} \mathcal{P}_{l_v} \mathcal{O}_{l_v}}_{\text{Decoding succ.}} + \underbrace{(1 - \sum_{v=1}^{T} \mathcal{P}_{l_v}) \mathcal{O}_{l_T}}_{\text{Decoding fail.}}.$$

- $\mathcal{O}_{\text{PASD}}$ is now channel dependent!

IV. Complexity Analysis

- *Theorem 2:* The decoding complexity of running the PASD algorithm with an OLS of l_v is:

$$\mathcal{O}_{l_v} = O(\mathcal{C}^2(\mathrm{M}_v)(l_v+1)).$$

where $\mathcal{C}(\mathrm{M}) = |\Lambda(\mathrm{M})| = \frac{1}{2}\sum_{i=0}^{q-1}\sum_{j=0}^{n-1} m_{ij}(m_{ij}+1)$

- $\mathcal{O}_{l_v}$ consists of $\mathcal{O}_{l_v}^{\text{int}}$ and $\mathcal{O}_{l_v}^{\text{fac}}$

$$\mathcal{O}_{l_v}^{\text{int}} = O(\mathcal{C}(\mathrm{M}_v)(\mathcal{C}(\mathrm{M}_v)+1)(l_v+1)) \cong O(\mathcal{C}^2(\mathrm{M}_v)(l_v+1))$$

$$\mathcal{O}_{l_v}^{\text{fac}} = O(k\sum_{\eta=1}^{v}(\mathcal{C}(\mathrm{M}_\eta)+1)l_\eta) < O(kv(\mathcal{C}(\mathrm{M}_v)+1)l_v)$$

since $kv \ll \mathcal{C}(\mathrm{M}_v)$

$$\mathcal{O}_{l_v}^{\text{fac}} \ll \mathcal{O}_{l_v}^{\text{int}}$$

$$\mathcal{O}_{l_v} \cong \mathcal{O}_{l_v}^{\text{int}}$$

IV. Complexity Analysis

- *Corollary 3:* When l_v is sufficiently large, the decoding complexity $\mathcal{O}_{l_v}$ becomes:

$$\mathcal{O}_{l_v} = O\left(\frac{(k-1)^2}{4}(l_v^5 + l_v^4)\right)$$

- The decoding complexity increases exponentially with the OLS (dominant exponential factor of 5);

- The decoding complexity is quadratic in the dimension of the code *k*. The decoding complexity will be smaller for a low rate code.

IV. Complexity Analysis

- *Theorem 4:* The probability of the PASD algorithm performing a successful decoding with an OLS of l_v is given as:

$$\mathcal{P}_{l_v} = \Pr[S_{\mathrm{M}_v}(\bar{c}) > \Delta_{1,k-1}(\mathcal{C}(\mathrm{M}_v)) \text{ and } S_{\mathrm{M}_{v-1}}(\bar{c}) \leq \Delta_{1,k-1}(\mathcal{C}(\mathrm{M}_{v-1}))]$$

- The probability that the algorithm terminates at iteration *v*.

Iter. 1: $\boldsymbol{S}_{\mathrm{M}1}(\mathbf{c}) \leq \Delta(C(\mathrm{M}_1))$

Iter. 2: $\boldsymbol{S}_{\mathrm{M}2}(\mathbf{c}) \leq \Delta(C(\mathrm{M}_2))$

$\vdots$

Iter. *v*-1: $\boldsymbol{S}_{\mathrm{M}v\text{-}1}(\mathbf{c}) \leq \Delta(C(\mathrm{M}_{v\text{-}1}))$

Iter. *v*: $\boldsymbol{S}_{\mathrm{M}v}(\mathbf{c}) > \Delta(C(\mathrm{M}_v)) \longrightarrow$ Exit!

IV. Complexity Analysis

- Determining a closed form expression of $\mathcal{P}_{l_v}$ turns out to be *hard*;

- Motivation of the analysis: link $\mathcal{P}_{l_v}$ with Π (~channel SNR);

- Define $\mathcal{L} = \left\{ l \;\middle|\; l > \frac{2 + \frac{\sqrt{n\gamma}}{\sqrt{\sum_{i,j} \pi_{ij}^2}}}{2\gamma[1 - \sqrt{k-1}\Phi_{\Pi}(\bar{c})]} \right\}$, where $\gamma = \frac{k-1}{n}$ and $\Phi_{\Pi}(\bar{c}) = \frac{\sqrt{\sum_{i,j} \pi_{ij}^2}}{S_{\Pi}(\bar{c})}$,

  with OLS greater than the above threshold, successful decoding can be guaranteed.

- We therefore interpret $\mathcal{P}_{l_v}$ as:

$$\mathcal{P}_{l_v} \cong \Pr[l_v = \min \mathcal{L}].$$

IV. Complexity Analysis

- Study the possible quantization of the OLS threshold $\dfrac{2+\dfrac{\sqrt{n\gamma}}{\sqrt{\sum_{i,j}\pi_{ij}^2}}}{2\gamma[1-\sqrt{k-1}\Phi_\Pi(\bar{c})]}$

- AWGN channel, we vary the SNR …

  In case of a 'bad' channel (e.g., SNR → -∞): $\pi_{ij} \cong 1/q$ for all $\pi_{ij} \in \Pi$

  $$\sqrt{\sum_{i,j}\pi_{ij}^2} \cong \sqrt{\tfrac{n}{q}} \quad S_\Pi(\bar{c}) \cong \tfrac{n}{q} \quad \Phi_\Pi(\bar{c}) \cong \sqrt{\tfrac{q}{n}}$$

  In case of a 'good' channel (e.g., SNR → + ∞): $\pi_{ij} \cong 1$ if $i = i_j$, and $\pi_{ij} \cong 0$ otherwise

  $$\sqrt{\sum_{i,j}\pi_{ij}^2} \cong \sqrt{n} \quad S_\Pi(\bar{c}) \cong n \quad \Phi_\Pi(\bar{c}) \cong \tfrac{1}{\sqrt{n}}$$

- By refining $\dfrac{1}{\sqrt{n}} \leq \Phi_\Pi(\bar{c}) < \dfrac{1}{\sqrt{k-1}}$,

  we can see the OLS threshold is a decreasing function of SNR.

- *Recall* $\mathcal{P}_{l_v} \cong \Pr[l_v = \min \mathcal{L}]$, $\mathcal{L} = \left\{ l \;\middle|\; l > \dfrac{2+\dfrac{\sqrt{n\gamma}}{\sqrt{\sum_{i,j}\pi_{ij}^2}}}{2\gamma[1-\sqrt{k-1}\Phi_\Pi(\bar{c})]} \right\}$

  $\mathcal{P}_{l_v}$ will be in favor of smaller l_v values by increasing SNR!

IV. Complexity Analysis

- *Recall* the average decoding complexity definition:

$$\mathcal{O}_{\text{PASD}} = \sum_{v=1}^{T} \mathcal{P}_{l_v} \mathcal{O}_{l_v} + (1 - \sum_{v=1}^{T} \mathcal{P}_{l_v}) \mathcal{O}_{l_T}.$$

In a sufficiently 'good' channel:

$\mathcal{P}_{l_1}$ → 1 and $\mathcal{O}_{\text{PASD}} \cong \mathcal{O}_{l_1}$

In a 'bad' enough channel:

$\mathcal{P}_{l_1}$ $\mathcal{P}_{l_2}$ … → 0 and $\mathcal{O}_{\text{PASD}} \cong \mathcal{O}_{l_T}$

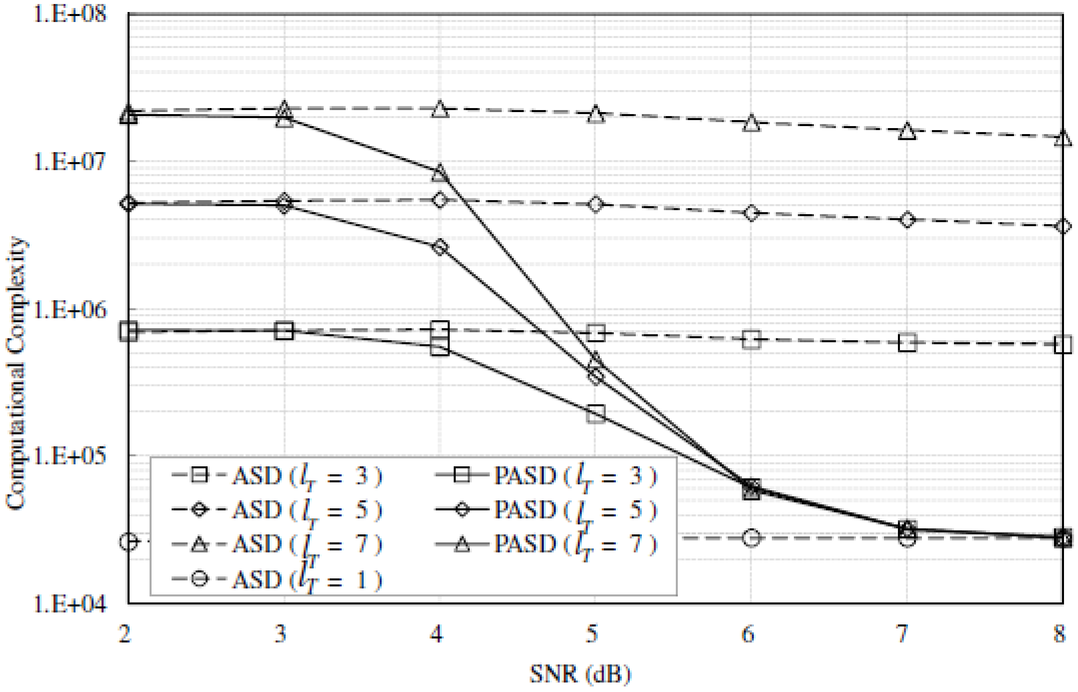
IV. Simulation Statistics

- $\mathcal{P}_{l_v}$ ~ SNR for the (63, 47) RS code with $l_1 = 1$, $l' = 1$ and $l_T = 5$
- AWGN channel

SNR (dB) / $\mathcal{P}_{l_v}$ (%)	2	3	4	5	6	7	8
$\mathcal{P}_{l_1}$	0.00	0.00	3.71	32.54	78.55	97.13	99.87
$\mathcal{P}_{l_2}$	0.00	1.51	28.47	56.30	21.25	2.87	0.13
$\mathcal{P}_{l_3}$	0.00	1.61	15.92	6.59	0.17	0.00	0.00
$\mathcal{P}_{l_4}$	0.00	1.23	9.68	1.95	0.02	0.00	0.00
$\mathcal{P}_{l_5}$	0.00	1.22	5.54	0.77	0.00	0.00	0.00

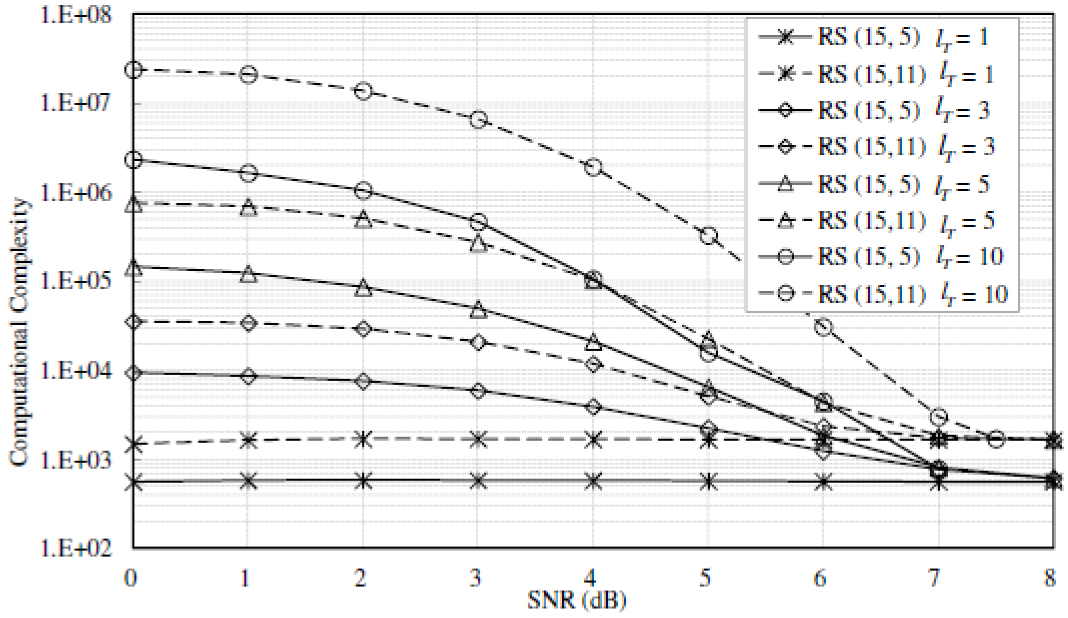
IV. Simulation Results

- $\mathcal{O}_{\text{PASD}}$ ~ SNR for the (63, 47) RS code with l_T = 3, 5, 7.
- AWGN channel



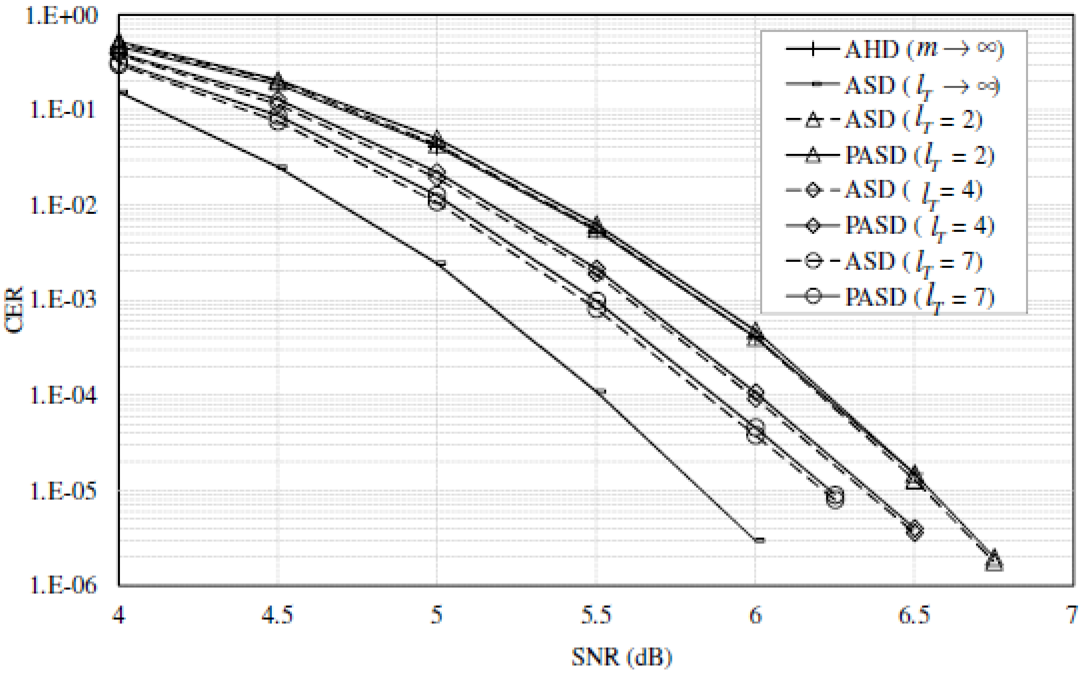
IV. Simulation Results

- $\mathcal{O}_{\text{PASD}}$ ~ SNR for the (15, 11) and (15, 5) RS codes with l_T = 1, 3, 5, 10
- AWGN channel



V. Error-Correction Performance

- PASD ~ ASD with same decoding parameter l_T;
- For PASD, decoding output is validated by CRC code;
- For ASD, decoding output is validated by the ML criterion;
- For the (63, 47) RS code, over AWGN channel:



VI. Conclusions

- A progressive algebraic soft-decision decoding approach;
- Two key steps of PASD: progressive reliability transform & progressive interpolation;
- Enables the complexity dominant interpolation process to be performed in an iterative manner, and the incremental computation between iterations is possible;
- The average decoding complexity of the PASD algorithm is channel dependent and hence it has been optimized according to the needs;
- Error-correction performance is also preserved.
- A larger system memory is required.

Acknowledgement

- Project: Advanced coding technology for future storage devices;
  ID: 61001094; From 2011. 1 to 2013. 12.

- Also funded by the Guangdong Natural Science Foundation (GNSF).
  Project: Research on adaptable list decoding system;
  ID: 10451027501005078; From 2010. 10 to 2012. 10.