A New Progressive Algebraic Soft Decoding Algorithm for Reed-Solomon Codes

Yi Lyu and Li Chen

School of Information Science and Technology, Sun Yat-sen University, Guangzhou, China
Email: lvyi3@mail2.sysu.edu.cn, chenli55@mail.sysu.edu.cn

***Abstract*—The progressive algebraic soft decoding (PASD) algorithm can leverage the average complexity for algebraic soft decoding (ASD) of Reed-Solomon (RS) codes. With a progressively enlarged decoding parameter that is the designed factorization output list size (OLS), it adapts the expensive interpolation computation to the quality of the received information and makes the average complexity of multiple decoding events channel dependent. However, the complexity reduction is realized at the expense of system memory since the intermediate interpolation information needs to be stored. Addressing this issue, this paper proposes a new PASD algorithm that can significantly reduce the memory requirement through the establishment of a condition on expanding the interpolated polynomial group without using the intermediate information. It has also embraced the interpolation coordinate transform (ICT) that alleviates the iterative polynomial construction task, resulting in the new proposal less computationally expensive than its predecessor, the PASD algorithm. Our numerical analysis shows that its memory requirement will be at most half of that of the PASD algorithm and it is less complex than various ASD algorithms, while the error-correction capability of ASD is preserved.**

***Index Terms*—Algebraic soft decoding, decoding complexity, progressive decoding, Reed-Solomon codes**

I. Introduction

Reed-Solomon (RS) codes are widely used in communications and storage systems. The conventional unique decoding algorithms [1][2] are efficient, but with limited error-correction capability. The algebraic hard decoding (AHD) algorithm [3] performs a curve-fitting decoding process and is able to correct errors beyond the code's half distance bound. The algebraic soft decoding (ASD) algorithm [4] matures the AHD approach by utilizing the soft information provided by the channel as it maps the reliability information to the interpolation multiplicity information. It outperforms the conventional unique decoding and the AHD algorithms.

In algebraic decodings, interpolation that is an iterative polynomial construction process dominates the computational complexity, and various complexity reduction approaches have been proposed so far. In [5], interpolation complexity is reduced by eliminating the polynomials with a leading order greater than the number of interpolation constraints. The interpolation complexity can also be reduced by utilizing the unique decoding outcome [6], which leads to a reduction of the interpolation multiplicity. In [7], interpolation complexity is reduced by performing a Chase-type algebraic decoding which saves computation by exploiting the similarity among several interpolation test-vectors. The interpolation coordinate transform (ICT) [8] that is realized through re-encoding a received word is another important complexity reduction approach. By transforming certain interpolation points into having a zero y-coordinate, the iterative polynomial construction task can be alleviated. Motivated by the fact that different decoding event may require different error-correction capability, the progressive ASD (PASD) [9] algorithm was recently proposed. It performs ASD with a progressively enlarged factorization output list size (OLS), adapting both the error-correction capability and computational complexity to the quality of the received information. Consequently, the average decoding complexity of multiple decoding events can be leveraged and becomes channel dependent. However, this approach is realized at the expense of system memory since the intermediate interpolation information needs to be stored as they will be utilized later in the progressive interpolation.

This paper proposes a new PASD algorithm that is capable of reducing the memory requirement and further the decoding complexity. The progressive interpolation can be considered as a progressive polynomial group expansion process. In this paper, a condition on expanding the polynomial group will be established such that the newly introduced polynomial into the group does not need to perform interpolation w.r.t. the previous constraints. Consequently, the memory requirement can be reduced significantly. Our numerical analysis shows that the new proposal reduces the memory requirement over its predecessor, the PASD algorithm, by a factor that is great than two. It also results in less computation for updating the newly introduced polynomial. Together with the ICT, the new PASD algorithm reduces the decoding complexity over the PASD algorithm by a factor of approximately three and meanwhile it is less complex than various ASD approaches. We will also confirm that this economic realization of ASD preserves its error-correction capability.

II. Preliminaries

Let $\mathbb{F}_q = \{\alpha_0, \alpha_1, ..., \alpha_{q-1}\}$ denote the finite field of size q, and $\mathbb{F}_q[x]$ and $\mathbb{F}_q[x, y]$ denote the univariate and bivariate polynomial rings defined over $\mathbb{F}_q$, respectively. To encode an (n, k) RS code, the message vector $\underline{u} = (u_0, u_1, \cdots, u_{k-1}) \in \mathbb{F}_q^k$ can be written as a polynomial $u(x) = u_0 + u_1x + \cdots + u_{k-1}x^{k-1}$, and the codeword $\underline{c} \in \mathbb{F}_q^n$ can be generated by

$$\underline{c} = (c_0, c_1, ..., c_{n-1}) = (u(x_0), u(x_1), ..., u(x_{n-1})), \quad (1)$$

where $x_0, x_1, ..., x_{n-1}$ are n distinct nonzero elements of $\mathbb{F}_q$. The code's minimum Hamming distance is $d = n - k + 1$.

It is assumed that an RS codeword is modulated and transmitted through a memoryless channel, e.g., the additive white Gaussian noise (AWGN) channel. Given a received vector $\underline{\mathcal{Y}} \in \mathbb{R}$, the $q \times n$ reliability matrix $\mathbf{\Pi}$ can be obtained with entry π_{ij} being defined as:

$$\pi_{ij} = \Pr[c_j = \alpha_i \mid \underline{\mathcal{Y}}]. \tag{2}$$

Matrix $\mathbf{\Pi}$ is then transformed into a multiplicity matrix $\mathbf{M}$ of the same size [4]. Entry m_{ij} represents the interpolation multiplicity for point (x_j, α_i). Interpolation is to iteratively construct a polynomial group $\mathbf{G}$ and each of its polynomials interpolates all the points defined by the nonzero entries of $\mathbf{M}$. Given an interpolated polynomial $Q = \sum_{a,b} Q_{ab} x^a y^b \in \mathbb{F}_q[x, y]$, for a nonnegative integer pair (r, s), the (r, s)th Hasse derivative evaluation of Q at point (x_j, α_i) is defined as [10]:

$$D_{r,s}(Q(x,y))|_{(x_j,\alpha_i)} = \sum_{a\geq r, b\geq s} \binom{a}{r}\binom{b}{s} Q_{ab} x_j^{a-r} \alpha_i^{b-s}. \tag{3}$$

Polynomial Q interpolates point (x_j, α_i) with a multiplicity of m_{ij} if $D_{r,s}(Q(x,y))|_{(x_j,\alpha_i)} = 0$ for all $r + s < m_{ij}$. For simplicity, we will use $(r, s)_{ij}$ to denote the interpolation constraints and Hasse derivative evaluation of (3) is denoted by $D_{(r,s)_{ij}}(Q)$.

In decoding an (n, k) RS code, monomials $x^a y^b$ are organized by the $(1, k-1)$-revlex order [5]. Given a polynomial $Q \in \mathbb{F}_q[x, y]$, if $x^{a'} y^{b'}$ is the leading monomial (lm) with coefficient $Q_{a'b'} \neq 0$, the $(1, k-1)$-weighted degree of Q is defined as $\deg_{1,k-1} Q = \deg_{1,k-1} \text{lm}(Q) = a' + (k-1)b'$ and its leading order is defined as $\text{lod}(Q) = \text{ord}(x^{a'} y^{b'})$. Given two polynomials $(H, Q) \in \mathbb{F}_q[x, y]$, we declare $H < Q$ if $\text{lod}(H) < \text{lod}(Q)$.

In matrix $\mathbf{M}$, let i_j denote the row index i with $\alpha_i = c_j$, i.e. $i_j = \{i \mid \alpha_i = c_j\}$. Entry $m_{i_j j}$ is the multiplicity for point (x_j, c_j). The codeword score is defined as $S_{\mathbf{M}}(\underline{c}) = \sum_{j=0}^{n-1} m_{i_j j}$. Given an interpolated polynomial Q, if [4]

$$S_{\mathbf{M}}(\underline{c}) > \deg_{1,k-1} Q, \tag{4}$$

the message polynomial $u(x)$ can be determined by factorization that finds out the y-roots of Q [11]. It delivers a list of message polynomial candidates $p(x)$ as:

$$\mathcal{L} = \{p(x) \in \mathbb{F}_q[x] \mid Q(x, p(x)) = 0 \text{ and } \deg_x p(x) < k\}. \tag{5}$$

The cardinality of $\mathcal{L}$ is referred as the factorization OLS.

The PASD algorithm performs decoding with a progressively enlarged designed OLS, adapting both decoding capability and computation to the actual need. In order to better explain the progressive decoding concept, the following definitions are introduced.

Definition I: Let $\Lambda(m_{ij})$ denote the set of interpolation constraints defined by a nonzero entry m_{ij} as:

$$\Lambda(m_{ij}) = \{(r, s)_{ij}, \forall\ r + s < m_{ij}\}. \tag{6}$$

$\Lambda(\mathbf{M})$ is used to denote a collection of all the constraint sets $\Lambda(m_{ij})$ that are defined by the nonzero entries of $\mathbf{M}$ as:

$$\Lambda(\mathbf{M}) = \{\Lambda(m_{ij}), \forall\ m_{ij} \in \mathbf{M} \text{ and } m_{ij} \neq 0\}. \ \square \tag{7}$$

The total number of constraints defined by matrix $\mathbf{M}$ is given by $\mathcal{C}(\mathbf{M}) = |\Lambda(\mathbf{M})| = \frac{1}{2} \sum_{i=0}^{q-1} \sum_{j=0}^{n-1} m_{ij}(m_{ij} + 1)$.

Definition II: Let $\mathbf{M}_A$ and $\mathbf{M}_B$ denote two multiplicity matrices of the same size with entries $m_{ij}^{(A)}$ and $m_{ij}^{(B)}$, respectively. With $m_{ij}^{(A)} \leq m_{ij}^{(B)}$ for all entries, the incremental interpolation constraints introduced by the two matrices are

$$\Lambda(\mathbf{M}_{B\backslash A}) = \{\Lambda(\mathbf{M}_B) \setminus \Lambda(\mathbf{M}_A)\}. \ \square \tag{8}$$

Note that $|\Lambda(\mathbf{M}_{B\backslash A})| = \mathcal{C}(\mathbf{M}_B) - \mathcal{C}(\mathbf{M}_A)$.

The PASD algorithm performs decoding with a series of monotonically increasing designed factorization OLS $l_1, l_2, \cdots, l_{v-1}, l_v, \cdots, l_T$, where $l_v = l_{v-1} + 1$ and l_T is the maximal OLS set according to the system's decoding budget. Correspondingly, a series of multiplicity matrices are generated as $\mathbf{M}_1, \mathbf{M}_2, \cdots, \mathbf{M}_{v-1}, \mathbf{M}_v, \cdots, \mathbf{M}_T$, where $m_{ij}^{(v-1)} \leq m_{ij}^{(v)}$ for all the matrices. It can be realized that

$$\Lambda(\mathbf{M}_v) = \Lambda(\mathbf{M}_1) \cup \Lambda(\mathbf{M}_{2\backslash 1}) \cup \cdots \cup \Lambda(\mathbf{M}_{v\backslash v-1}) \tag{9}$$

for $v = 1, 2, \ldots, T$. Note that $\Lambda(\mathbf{M}_0) = [0]_{q\times n}$. The PASD algorithm is to perform interpolation w.r.t. the constraints of $\Lambda(\mathbf{M}_1)$, $\Lambda(\mathbf{M}_{2\backslash 1})$, $\ldots$, $\Lambda(\mathbf{M}_{v\backslash v-1})$ progressively. If Q_v is the interpolated polynomial that has satisfied all the constraints of $\Lambda(\mathbf{M}_v)$, the intended message polynomial can be found if [9]

$$S_{\mathbf{M}_v}(\underline{c}) > \deg_{1,k-1}(Q_v), \tag{10}$$

and the decoding will be terminated. Note that $S_{\mathbf{M}_v}(\underline{c})$ is defined similarly as $S_{\mathbf{M}}(\underline{c})$. Therefore, in a good channel condition most of the received information are mildly corrupted, condition of (10) will happen in an earlier decoding stage with a smaller OLS value, and vice versa. Since the decoding complexity grows exponentially with the OLS [9], average decoding complexity of the new PASD algorithm is channel dependent.

III. The New Progressive Interpolation

The new PASD algorithm embraces two important features into its progressive interpolation to reduce the memory requirement and decoding complexity. They are the ICT that is performed based on $\mathbf{M}_1$ and the new progressive polynomial group expansion.

A. The ICT Based on $\mathbf{M}_1$

The set of interpolation points indicated by matrix $\mathbf{M}_1$ can be defined as $P_{\mathbf{M}_1} = \{(x_j, \alpha_i),\ \forall\ m_{ij} \in \mathbf{M}_1 \text{ and } m_{ij} \neq 0\}$. We first identify k points in the set $P_{\mathbf{M}_1}$ to perform re-encoding [8]. In matrix $\mathbf{\Pi}$, the maximal entry of each column can be identified as $\pi_j^* = \max\{\pi_{ij} \mid i = 0, 1, \cdots, q-1\}$. Sorting the n maximal entries in a descending order yields a refreshed column index sequence $\theta_0, \theta_1, \cdots, \theta_{k-1}, \cdots, \theta_{n-1}$, which implies $\pi_{\theta_0}^* \geq \pi_{\theta_1}^* \geq \cdots \geq \pi_{\theta_{k-1}}^* \geq \cdots \geq \pi_{\theta_{n-1}}^*$. Let $i(\theta)$ denote the row index of entry π_θ^* as $i(\theta) = \{i \mid \pi_{i,\theta} = \pi_\theta^*\}$, the following set of points can be constituted for re-encoding as $P_{\mathbf{M}_1}^{\text{I}} = \{(x_{\theta_0}, \alpha_{i(\theta_0)}), (x_{\theta_1}, \alpha_{i(\theta_1)}), \cdots, (x_{\theta_{k-1}}, \alpha_{i(\theta_{k-1})})\}$ and $|P_{\mathbf{M}_1}^{\text{I}}| = k$. Let $\Theta = \{\theta_0, \theta_1, \cdots, \theta_{k-1}\}$, the re-encoding polynomial $T(x)$ can be defined as:

$$T(x) = \sum_{j \in \Theta} \alpha_{i(j)} t_j(x), \tag{11}$$

where $t_j(x) = \prod_{\substack{(j,\delta)\in\Theta,\\ \delta\neq j}} \frac{x - x_\delta}{x_j - x_\delta}$ is the Lagrange basis polynomial and $T(x_j) = \alpha_{i(j)}$ for all $j \in \Theta$. Therefore, for the original interpolation points (x_j, α_i), we can transform them by $(x_j, \alpha_i + T(x_j))$. Consequently, set $P_{\mathbf{M}_1}$ is transformed into $P'_{\mathbf{M}_1} = \{(x_j, \alpha_i + T(x_j)),\ \forall\ m_{ij} \in \mathbf{M}_1$ and $m_{ij} \neq 0\}$. In particular, $P^{\mathrm{I}}_{\mathbf{M}_1}$ is transformed into a set of points with a zero y-coordinate as $P^{\mathrm{I}'}_{\mathbf{M}_1} = \{(x_j, 0),\ \forall\ j \in \Theta\}$. Note that in matrix $\mathbf{M}_1$, there can be less than k nonzero entries. The above sorting process aims to enable set $P^{\mathrm{I}}_{\mathbf{M}_1}$ (or $P^{\mathrm{I}'}_{\mathbf{M}_1}$) include most of the points that correspond to the nonzero entries of $\mathbf{M}_1$, so that the following process can reduce the interpolation complexity in its best capability.

At the beginning, with an initial factorization OLS $l_1 = 1$, polynomial group $\mathbf{G}_1 = \{g_0, g_1\}$ is initialized by [8]

$$g_u = y^u \prod_{j \in \Theta} (x - x_j)^{[m_{i(j)j} - u]^+}, \tag{12}$$

where $u = 0, 1$ and $[m_{i(j)j} - u]^+ = \max\{m_{i(j)j} - u, 0\}$. Let $\Lambda^{\mathrm{I}}(\mathbf{M}_1)$ denote the set of interpolation constraints defined by points of $P^{\mathrm{I}'}_{\mathbf{M}_1}$, polynomials of $\mathbf{G}_1$ have already satisfied those constraints. They will further perform interpolation w.r.t. the remaining constraints of $\{\Lambda(\mathbf{M}_1) \setminus \Lambda^{\mathrm{I}}(\mathbf{M}_1)\}$. Regarding each constraint $(r, s)_{ij} \in \{\Lambda(\mathbf{M}_1) \setminus \Lambda^{\mathrm{I}}(\mathbf{M}_1)\}$, Hasse derivative evaluation will be performed for each polynomial of $\mathbf{G}_1$. For those polynomials with $D_{(r,s)_{ij}}(g_u) \neq 0$, the minimal one will be selected as:

$$f = \min\{g_u \mid D_{(r,s)_{ij}}(g_u) \neq 0\}, \tag{13}$$

and the bilinear modification will be performed following [4]:

$$g_u = \begin{cases} g_u - \dfrac{D_{(r,s)_{ij}}(g_u)}{D_{(r,s)_{ij}}(f)} f, & \text{if } g_u \neq f, \quad (14a) \\ (x - x_j) f, & \text{if } g_u = f. \quad (14b) \end{cases}$$

We define the above bilinear modification as the *polynomial update* such that the updated polynomial satisfies the current constraint $(r, s)_{ij}$. After interpolation w.r.t. constraints of $\{\Lambda(\mathbf{M}_1) \setminus \Lambda^{\mathrm{I}}(\mathbf{M}_1)\}$ has been performed, the minimal polynomial in $\mathbf{G}_1$ will be chosen as Q_1 for factorization. If $u'(x) \in \mathbb{F}_q[x]$ with $\deg_x u'(x) < k$ is the factorization outcome, the intended message polynomial can be further determined by $u(x) = u'(x) + T(x)$. If the intended message polynomial cannot be found [1], the OLS will be increased to l_2 forcing another stronger ASD attempt.

B. New Progressive Polynomial Group Expansion

In the PASD algorithm, the OLS increment will immediately lead to the polynomial group expansion, i.e., $|\mathbf{G}_v| = l_v + 1$ where $\mathbf{G}_v$ is the polynomial group of progressive iteration v. The newly introduced polynomial will have to perform re-interpolation w.r.t. the previous constraints by engaging with the identified minimal polynomials f as in ($14a$). It requires a large memory for storing those polynomials. The new proposal significantly reduces the memory requirement by establishing a new polynomial group expanding condition with which most of the re-interpolation can be excepted. The following description formulates the establishment of the condition.

Let σ denote the index of an interpolation constraint $(r, s)_{ij}$ and kernel $\mathcal{K}_\sigma$ can be defined as:

$$\mathcal{K}_\sigma = \{Q \in \mathbb{F}_q[x, y] \mid D_{(r,s)_{ij}}(Q) = 0\}. \tag{15}$$

Accumulated kernel $\overline{\mathcal{K}}_\sigma$ can be further defined as $\overline{\mathcal{K}}_\sigma = \mathcal{K}_\sigma \cap \overline{\mathcal{K}}_{\sigma-1} = \mathcal{K}_\sigma \cap \mathcal{K}_{\sigma-1} \cap \cdots \cap \mathcal{K}_1$. We also define $\mathcal{W}_b$ as:

$$\mathcal{W}_b = \{Q \in \mathbb{F}_q[x, y] \mid \deg_y \mathrm{lm}(Q) = b\}. \tag{16}$$

In the ASD algorithm [4] that performs decoding with an OLS of l_T, the interpolation begins with a polynomial group $\mathbf{G}^{(0)} = \{g_0^{(0)} = 1, g_1^{(0)} = y, \cdots, g_u^{(0)} = y^u, \cdots, g_{l_T}^{(0)} = y^{l_T}\}$, and each of its polynomials $g_u^{(0)}$ is the minimal polynomial of $\mathcal{W}_u$ where $u = 0, 1, \ldots, l_T$. After the σth constraint has been satisfied, group $\mathbf{G}^{(0)}$ evolves to $\mathbf{G}^{(\sigma)} = \{g_0^{(\sigma)}, g_1^{(\sigma)}, \cdots, g_u^{(\sigma)}, \cdots, g_{l_T}^{(\sigma)}\}$ [2] and each of its polynomials $g_u^{(\sigma)} = \min\{\overline{\mathcal{K}}_\sigma \cap \mathcal{W}_u\}$. By observing $g_{u+1}^{(0)}$ is divisible by $g_u^{(0)}$, the following lemma can be led to.

Lemma 1: If $\mathrm{lod}(g_u^{(\sigma)}) = \mathrm{lod}(g_u^{(0)})$, then $y g_u^{(\sigma)}$ can be the candidate of $g_{u+1}^{(\sigma)}$ [13].

Lemma 1 implies as far as $\mathrm{lod}(g_u^{(\sigma)}) = \mathrm{lod}(g_u^{(0)})$, $y g_u^{(\sigma)} = \min\{\overline{\mathcal{K}}_\sigma \cap \mathcal{W}_{u+1}\}$. Notice that during the polynomial update, only ($14b$) will lead to the lod increase. That says as far as $g_u^{(\sigma)}$ has not become the minimal polynomial f, $g_{u+1}^{(\sigma)}$ can always be generated by $g_{u+1}^{(\sigma)} = y g_u^{(\sigma)}$. Therefore, the following theorem establishes the expanding condition and the updating operation for the newly introduced polynomial.

Theorem 2: In a polynomial group $\mathbf{G}^{(\sigma)}$ with $|\mathbf{G}^{(\sigma)}| = u + 1$, if $g_u^{(\sigma)}$ is the minimal polynomial that does not satisfy the current constraint $(r, s)_{i,j}$, polynomial $g_{u+1}^{(\sigma+1)}$ should be introduced into the group and updated by [13]

$$g_{u+1}^{(\sigma+1)} = (y - \alpha_i) g_u^{(\sigma)}, \tag{17}$$

such that it satisfies all the interpolation constraints that polynomial $g_u^{(\sigma)}$ has satisfied and the current one.

Proof: Based on Lemma 1, we know that the new polynomial $g_{u+1}^{(\sigma)}$ shall be introduced into the group as $g_{u+1}^{(\sigma)} = y g_u^{(\sigma)}$ so that it satisfies all the constraints that $g_u^{(\sigma)}$ has satisfied. To further enable polynomial $g_{u+1}^{(\sigma+1)}$ satisfy the current constraint, update of ($14a$) will be performed as:

$$\begin{aligned} g_{u+1}^{(\sigma+1)} &= g_{u+1}^{(\sigma)} - \frac{D_{(r,s)_{ij}}(g_{u+1}^{(\sigma)})}{D_{(r,s)_{ij}}(g_u^{(\sigma)})} g_u^{(\sigma)} \\ &= y g_u^{(\sigma)} - \frac{\alpha_i D_{(r,s)_{ij}}(g_u^{(\sigma)})}{D_{(r,s)_{ij}}(g_u^{(\sigma)})} g_u^{(\sigma)} \\ &= (y - \alpha_i) g_u^{(\sigma)}. \ \blacksquare \end{aligned}$$

[1] The decoding output validation is performed by the maximum likelihood criterion of [12].

[2] In the progressive interpolation, there are two types of iterations. One is the iterative polynomial construction for which we use σ to denote the index of an interpolation constraint and $\mathbf{G}^{(\sigma)}$ to denote the polynomial group w.r.t. the constraint. The other is the progressive iteration for which we use v to denote the iteration index and $\mathbf{G}_v$ to denote the polynomial group of progressive iteration v.

The above description shows the polynomial group does not expand simultaneously with the designed OLS increment. For this reason, during the progressive iteration v, we use l'_v to denote the maximal y-degree of polynomials in $\mathbf{G}_v$ as:

$$l'_v = \max\{\deg_y g_u \mid g_u \in \mathbf{G}_v\}. \tag{18}$$

It implies $|\mathbf{G}_v| = l'_v + 1$. During the progressive iterations, $1 \leq l'_v \leq l_v \leq l_T$. It is assumed that l'_v also progresses by

$$l'_v = l'_{v-1} + 1. \tag{19}$$

Without loss of generality, we now describe the new PASD algorithm as being performed at iteration $v-1$ with a designed factorization OLS of l_{v-1}, where $2 < v \leq T$. At the beginning of iteration $v-1$, we have polynomial group

$$\mathbf{G}_{v-1} = \{g_0, g_1, \cdots, g_{l'_{v-1}}\}. \tag{20}$$

They all satisfy the constraints of $\Lambda(\mathbf{M}_{v-2})$. At the current iteration, they perform the iterative polynomial updates as in (13)-(14b) w.r.t. constraints of $\Lambda(\mathbf{M}_{v-1\setminus v-2})$. Based on Theorem 2, we know during the polynomial updates, once

$$f = g_{l'_{v-1}}, \tag{21}$$

we need to introduce a new polynomial g^* by:

$$g^* = (y - \alpha_i) g_{l'_{v-1}}, \tag{22}$$

where α_i is the y-coordinate of the current interpolation point. We can now determine whether g^* should be included in group $\mathbf{G}_{v-1}$ to participate into the interpolation w.r.t. the remaining constraints of $\Lambda(\mathbf{M}_{v-1\setminus v-2})$. The following two cases can be classified for the group expansion.

Case 1.1: If $l'_{v-1} < l_{v-1}$, polynomial group $\mathbf{G}_{v-1}$ needs to be expanded by

$$\mathbf{G}_{v-1} = \{g_0, g_1, \cdots, g_{l'_{v-1}}\} \cup \{g^*\}. \tag{23}$$

Afterwards, l'_{v-1} will be increased by one and $\mathbf{G}_{v-1}$ can be expressed as:

$$\mathbf{G}_{v-1} = \{g_0, g_1, \cdots, g_{l'_{v-1}-1}, g_{l'_{v-1}}\}. \tag{24}$$

Polynomials of $\mathbf{G}_{v-1}$ will perform interpolation w.r.t. the remaining constraints of $\Lambda(\mathbf{M}_{v-1\setminus v-2})$.

Case 1.2: If $l'_{v-1} = l_{v-1}$, polynomial group $\mathbf{G}_{v-1}$ does not need to be expanded, since the newly introduced polynomial g^* will not be chosen to be factorized after the current iteration. Instead, it will be stored in memory. Polynomials of $\mathbf{G}_{v-1}$ will continue to perform interpolation w.r.t. the remaining constraints $(r_1, s_1)_{i_1 j_1}, (r_2, s_2)_{i_2 j_2}, \cdots, (r_{\mathcal{L}}, s_{\mathcal{L}})_{i_{\mathcal{L}} j_{\mathcal{L}}}$ of $\Lambda(\mathbf{M}_{v-1\setminus v-2})$ and the identified minimal polynomials f will be stored.

In the end of iteration $v-1$, an updated polynomial group $\tilde{\mathbf{G}}_{v-1}$ will be yielded and

$$\tilde{\mathbf{G}}_{v-1} = \{\tilde{g}_0, \tilde{g}_1, \cdots, \tilde{g}_{l'_{v-1}}\}. \tag{25}$$

The minimal polynomial of $\tilde{\mathbf{G}}_{v-1}$ will be chosen as Q_{v-1} to be factorized. If the intended message polynomial cannot be found after the factorization, the OLS l_{v-1} will be increased by one and matrix $\mathbf{M}_v$ will be generated accordingly.

In progressive iteration v, polynomial group $\mathbf{G}_v$ inherits information from $\tilde{\mathbf{G}}_{v-1}$ and such a process can again be classified into two cases.

TABLE I
AVERAGE MEMORY REQUIREMENT IN DECODING THE (15, 11) RS CODE

SNR (dB)		2	3	4	5	6	7
$l_T = 5$	PASD	8213	6075	3060	945	202	8
	new PASD	2924	1980	876	232	30	0
$l_T = 7$	PASD	30258	20571	9664	2800	437	8
	new PASD	12035	8534	4008	1063	130	0

Case 2.1: If $l'_{v-1} + 1 < l_v$, it implies in the previous iteration, condition of (21) has not occurred or it has occurred with **Case 1.1**. Hence

$$\mathbf{G}_v = \tilde{\mathbf{G}}_{v-1}. \tag{26}$$

Case 2.2: If $l'_{v-1} + 1 = l_v$, it implies **Case 1.2** has occurred and an expansion is needed following

$$\mathbf{G}_v = \tilde{\mathbf{G}}_{v-1} \cup \{g^*\}. \tag{27}$$

Polynomial g^* will perform interpolation w.r.t. constraints of $(r_1, s_1)_{i_1 j_1}, (r_2, s_2)_{i_2 j_2}, \cdots, (r_{\mathcal{L}}, s_{\mathcal{L}})_{i_{\mathcal{L}} j_{\mathcal{L}}}$ by engaging with the corresponding minimal polynomials f as in (14a), yielding an updated polynomial $\tilde{g}^*$. Note that g^* may become the minimal polynomial f during the update. If so, it will be utilized to generate a new polynomial as in (22).

With defining the polynomial group $\mathbf{G}_v$, the following interpolation will be performed as in iteration $v-1$. The decoding will be terminated either when the intended message polynomial is found or the maximal OLS l_T is reached.

IV. NUMERICAL ANALYSIS

The above description shows that the new PASD algorithm does not need to store all the intermediate interpolation information. The intermediate information only needs to be stored in **Case 1.2**. Moreover, with performing ICT based on $\mathbf{M}_1$, there are at most $|\Lambda^{\mathrm{I}}(\mathbf{M}_1)|$ minimal polynomials f have been excepted from the storage. As a result, the new PASD algorithm's memory requirement will be less than that of the PASD algorithm. Table I shows the average memory requirement for storing the minimal polynomials f in decoding the (15, 11) RS code. It is measured in the AWGN channel with using BPSK for transmission. It is averaged over running 10 000 independent decoding events at each signal-to-noise ratio (SNR). It is assumed that one polynomial coefficient consumes one memory unit. Note that such a simulation testbed and measurement setup will also be adopted in the computational complexity analysis later. It can be seen that with the same maximal OLS, the new PASD algorithm requires less than half of the memory of its predecessor. As the SNR increases, the average memory requirement reduces since more decoding events will produce the intended outcome with a smaller OLS value. E.g. at 7dB, most of the intended message polynomials can be delivered after the first iteration in which only polynomial initialization of (12) has been performed. The new PASD algorithm does not store any intermediate information.

Table II shows the average number of finite field arithmetic operations in decoding the (15, 11) RS code. The data are presented as in $\times 10^4$. E.g., at 5dB, the average complexity

TABLE II
AVERAGE COMPUTATIONAL COMPLEXITY IN DECODING THE (15, 11) RS CODE WITH $l_T = 5$

SNR (dB)	2	3	4	5	6	7	8
ASD	208	208	200	181	159	141	127
ASD of [8]	89.0	77.0	54.0	35.0	23.0	18.0	15.0
PASD	134	96.0	47.0	14.0	2.54	0.77	0.72
new PASD	92.0	67.0	31.0	9.00	1.50	0.46	0.43

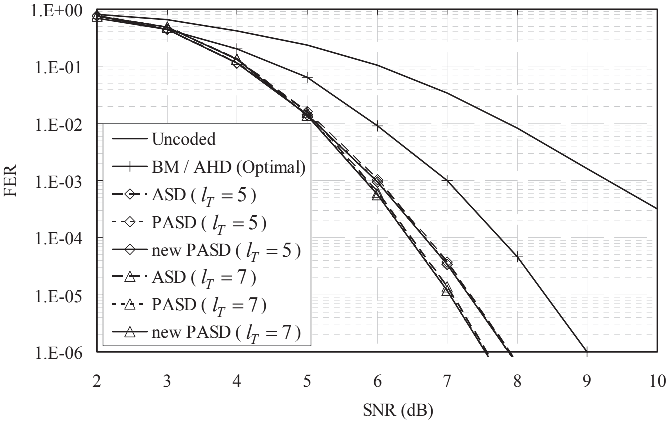

Fig. 1. FER performance of the (15, 11) RS code over the AWGN channel.

of the new PASD algorithm is 9×10^4. The comparison benchmarks include the original ASD algorithm of [4], the complexity reducing ASD algorithm of [8] and the PASD algorithm of [9]. All the algorithms are functioning with the same decoding budget as indicated by $l_T = 5$. For the ASD algorithm of [8] with a designed OLS of l_T, its ICT will be performed based on matrix $\mathbf{M}_T$. Table II shows the average complexity of the progressive decoding algorithms reduces significantly as the SNR increases and they are far simpler than the ASD algorithm. Compared to the PASD algorithm, the new proposal offers a complexity reduction by a factor of approximately three, thanking to both the ICT and the new approach for expanding the polynomial group. In the new expansion, tangible amount of updates for the newly introduced polynomials have been skipped. It can also be noticed that in the low SNR region, e.g. at 2-3dB, the ASD algorithm of [8] has a similar complexity as the new PASD algorithm. It is because in this region the new PASD algorithm delivers the intended message polynomial with a large OLS value. The ASD algorithm of [8] is also very competent in reducing the complexity by performing the ICT with matrix $\mathbf{M}_T$ as more iterative polynomial construction task can be replaced by polynomial initializations.

Finally, Fig.1 shows the frame error rate (FER) performance of the same RS code over the AWGN channel using BPSK. The algebraic soft decoding algorithms are functioning with the same l_T value. The unique decoding is performed by the Berlekamp-Massey (BM) algorithm [1] and the optimal AHD result is obtained by assuming it can correct at most $n - \lfloor \sqrt{n(n-d)} \rfloor - 1$ symbol errors. It shows the algebraic soft decoding algorithms outperform both the unique decoding and the AHD algorithms. In particular, the new PASD algorithm preserves the error-correction capability of the ASD algorithm, but with a far smaller average decoding complexity.

V. CONCLUSIONS

This paper has proposed a new PASD algorithm that can significantly reduce the average decoding complexity and the memory requirement. A new polynomial group expanding condition has been established such that during the progressive interpolation, the newly introduced polynomial does not need to perform re-interpolation w.r.t. the previous constraints. Further assisted by performing the ICT at the beginning of the progressive decoding, the new PASD algorithm is less complex than the PASD algorithm. Our numerical analysis has shown that the new proposal requires less than half of the memory that would be required by the PASD algorithm and it is less complex than various ASD approaches. Finally, we have confirmed the new PASD algorithm preserves the error-correction capability of the ASD algorithm.

ACKNOWLEDGEMENT

This work is sponsored by the National Basic Research Program of China (973 Program) with project ID 2012CB316100 and the National Natural Science Foundation of China (NSFC) with project ID 61372079.

REFERENCES

[1] J. Massey, “Shift register synthesis and BCH decoding,” *IEEE Trans. Inform. Theory*, vol. 15, no. 1, pp. 122-127, Jan. 1991.

[2] L. Welch and E. Berlekamp, “Error correction for algebraic block codes,” *Proc. IEEE Int. Symp. Inform. Theory*, Quebec, Canada, 1983.

[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.

[4] R. Koetter and A. Vardy, “Algebraic soft-decision decoding of Reed-Solomon codes,” *IEEE Trans. Inform. Theory*, vol. 49, no. 11, pp. 2809-2825, Nov. 2003.

[5] L. Chen, R. A. Carrasco and E. G. Chester, “Performance of Reed-Solomon codes using the Guruswami-Sudan algorithm with improved interpolation efficiency,” *IET Proc. Commun.*, vol. 1, no. 2, pp. 241-250, Apr, 2007.

[6] Y. Wu, “New list decoding algorithm for Reed-Solomon and BCH codes,” *IEEE Trans. Inform. Theory*, vol. 54, no. 8, pp. 3611-3630, Aug. 2008.

[7] J. Bellorado and A. Kavčić, “Low-complexity soft-decoding algorithms for Reed-Solomon codes—Part I: An algebraic soft-in hard-out Chase decoder,” *IEEE Trans. Inform. Theory*, vol. 56, no. 3, pp. 68-79, Mar. 2010.

[8] R. Koetter, J. Ma, and A. Vardy, “The re-encoding transformation in algebriac list-decoding of Reed-Solomon codes,” *IEEE Trans. Inform. Theory*, vol. 57, no. 2, pp. 633-647, Feb. 2011.

[9] L. Chen, S. Tang, and X. Ma, “Progressive algebraic soft-decision decoding of Reed-Solomon codes,” *IEEE Trans. Commun.*, vol. 61, no. 2, pp. 433-442, Feb. 2013.

[10] H. Hasse, “Theorie der hoheren Differentiale in einem algebraishen Funcktionenkorper mit vollkommenem konstantenkorper nei beliebeger Charakteristic,” *J. Reine. Aug. Math.*, pp. 50-54, 1936.

[11] R. M. Roth and G. Ruckenstein, “Efficient decoding of Reed-Solomon codes beyond half the minimum distance,” *IEEE Trans. Inform. Theory*, vol. 46, no. 1, pp. 246-256, Jan. 2000.

[12] T. Kaneko, T. Nishijima, H. Inazumi, and S. Hirasawa, “An efficient maximum-likelihood-decoding algorithm for linear block codes with algebraic decoder,” *IEEE Trans. Inform. Theory*, vol. 40, no. 2, pp. 320-327, Mar. 1994.

[13] Y. Zhu and S. Tang, “A reduced-complexity algorithm for polynomial interpolation,” *Proc. IEEE Int. Symp. Inform. Theory*, Istanbul, Turkey, 2013.