Progressive Algebraic Soft-Decision Decoding of Reed–Solomon Codes Using Module Minimization

Jiongyue Xing, *Student Member, IEEE*, Li Chen, *Senior Member, IEEE*, and Martin Bossert, *Fellow, IEEE*

***Abstract*— The algebraic soft-decision decoding (ASD) of Reed–Solomon (RS) codes yields a competent decoding performance with a polynomial-time complexity. But its complexity remains high due to the interpolation that generates the interpolation polynomial $Q(x, y)$. The progressive ASD (PASD) algorithm has been introduced to construct $Q(x, y)$ with a progressively enlarged y-degree, adjusting its error-correction capability and computation to the received information. However, this progressive decoding is realized at the cost of memorizing the intermediate decoding information. To overcome this challenge, this paper proposes a new PASD algorithm which is evolved from the ASD using module minimization (MM) interpolation. Polynomial $Q(x, y)$ can be constructed through the image of the progressively enlarged submodule basis without the need of memorizing the intermediate decoding information, eliminating the memory cost of progressive decoding. The MM interpolation also attributes to a remarkably lower complexity than the original PASD algorithm. Furthermore, a complexity reducing variant is proposed based on assessing the degree of Lagrange interpolation polynomials. We also analyze the complexity of the proposed decoding methods and reveal their channel dependent feature. Our simulation results show their low-complexity and advanced decoding performances.**

***Index Terms*— Algebraic soft-decision decoding, complexity reduction, module minimization, progressive interpolation, Reed–Solomon codes.**

I. Introduction

REED-SOLOMON (RS) codes are widely employed in communication systems and storage devices for error-correction. The conventional unique decoding algorithms such as the Berlekamp-Massey (BM) algorithm [1] [2] and the Welch-Berlekamp algorithm [3], can correct errors up to half of the code's minimum Hamming distance. Assisted by soft information, the generalized minimum-distance (GMD) decoding algorithm [4] improves the error-correction performance by performing the error-erasure decoding.

In late 90s, Sudan proposed the interpolation based algebraic decoding to correct errors beyond the half distance bound [5]. However, this only applies to codes of rate less than $\frac{1}{3}$. Later, Guruswami and Sudan improved it to decode all rate codes [6]. This is called the Guruswami-Sudan (GS) algorithm. It consists of two major steps, interpolation and root-finding. Interpolation finds the minimum polynomial $Q(x, y)$, which is often realized by Koetter's iterative polynomial construction approach [7]. Afterwards, root-finding determines the y-roots of $Q(x, y)$, which may contain the intended message [8]. By transforming soft received information into multiplicity information, Koetter and Vardy introduced the algebraic soft-decision decoding (ASD) algorithm [9], the so-called KV algorithm, that significantly outperforms its hard-decision counterpart, the GS algorithm. Several techniques are applied for this transform to achieve a better ASD performance [10] [11]. Also utilizing soft received information, the algebraic Chase decoding algorithm [12] constructs a number of test-vectors which share some common symbols. This allows the interpolation of all test-vectors to be performed in a binary tree growth fashion, resulting in a low decoding complexity. There also exist several approaches to reduce the complexity of Koetter's interpolation, including the re-encoding transform [13] [14] and the divide-and-conquer interpolation [15] [16]. Cassuto *et al.* further analyzed the dependence of interpolation cost on the error weights and proposed an interpolation algorithm that reduces the average-case decoding complexity [17].

Besides Koetter's interpolation, polynomial $Q(x, y)$ can also be determined using the concept of module and its basis reduction [18]. One can formulate a basis of module which contains bivariate polynomials that interpolate all the prescribed points with their multiplicity. Presenting the basis as a matrix over univariate polynomials, row operation further reduces it into the Gröbner basis [19] defined under a weighted monomial order. The minimum candidate of the basis is the intended polynomial $Q(x, y)$. This interpolation technique is called module minimization (MM) which is also referred as basis reduction in computer algebra. The basis reduction can be realized by the conventional Mulders-Storjohann (MS) algorithm [20]. Meanwhile, several asymptotically faster basis reduction approaches have been proposed in [21]–[24]. Lee and O'Sullivan presented an explicit module basis construction and reduction for the ASD,

Manuscript received October 16, 2018; revised March 26, 2019 and June 10, 2019; accepted June 26, 2019. Date of publication July 5, 2019; date of current version November 19, 2019. This work is sponsored by the National Natural Science Foundation of China (NSFC) with project ID 61671486 and International Program for Ph.D. Candidates, Sun Yat-sen University. This paper was presented in part at the 2018 IEEE International Symposium on Information Theory (ISIT). The associate editor coordinating the review of this paper and approving it for publication was P. Trifonov. *(Corresponding author: Li Chen.)*

J. Xing is with the School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou 510006, China (e-mail: xingjyue@mail2.sysu.edu.cn).

L. Chen is with the School of Electronics and Communication Engineering, Sun Yat-sen University, Guangzhou 510006, China (e-mail: chenli55@mail.sysu.edu.cn).

M. Bossert is with the Institute of Communications Engineering, Ulm University, 89081 Ulm, Germany (e-mail: martin.bossert@uni-ulm.de).

Digital Object Identifier 10.1109/TCOMM.2019.2927207

namely the ASD-MM algorithm [25]. Ma and Vardy further utilized the re-encoding transform to reduce the ASD-MM complexity [26]. The MM interpolation has also been generalized to perform the multi-trial GS (MT-GS) decoding [27], the algebraic Chase decoding [28] and power decoding [29]. Based on another structure of ideal, Trifonov proposed a fast randomized ideal multiplication algorithm to reduce the GS decoding complexity [30]. Its soft-decision and re-encoding transformed variants have been introduced in [31] and [32], respectively. Until now, the asymptotically fastest interpolation algorithms appear in [24] and [33].

For the above mentioned interpolation based algebraic decoding, the error-correction capability is determined by the y-degree of $Q(x, y)$, i.e., $\deg_y Q$. However, enlarging $\deg_y Q$ also implies a heavier decoding computation. In order to adjust the error-correction capability and decoding complexity to the received information, the progressive ASD (PASD) algorithm has been proposed in [34]. Utilizing Koetter's interpolation to construct $Q(x, y)$, it enlarges $\deg_y Q$ gradually, resulting in a progressively expanded polynomial set. It terminates once the intended message is found. As a result, when the received information is reliable as in high signal-to-noise ratio (SNR), the message can be decoded with the smallest parameter and the least computation effort. However, the polynomial set expansion requires knowledge of the intermediate decoding information. The progressive decoding is realized at the cost of system memory. Despite the later effort [35] to alleviate the memory cost, the PASD algorithm still exhibits a memory cost that is quadratic in the codeword length.

To overcome this challenge and further reduce the decoding complexity, this paper proposes a new PASD algorithm in which its progressive interpolation is realized by the MM technique. It is named the PASD-MM algorithm. This research shows utilizing the MM technique, the progressive interpolation can be realized through the image of the progressively enlarged submodule basis. To determine an interpolation polynomial $Q(x, y)$ with a larger y-degree, one could expand the image and further reduce it into the desired form. During the expansion, the newly introduced polynomial can be directly generated from the enumerated interpolation points. Consequently, the cost of memorizing the intermediate decoding information can be removed. We also show that a common multiplier of all entries of the submodule basis can be divided away, yielding an image with entries (univariate polynomials) of lower degrees. This results in a lower image expansion and reduction complexity. It should be pointed out that this work can be seen as a soft-decision decoding development of the earlier MT-GS algorithm [27]. The MT-GS algorithm decodes the message with progressively enlarged parameters, multiplicity and $\deg_y Q$, also resulting in a progressively expanded module basis without additional memory requirement. Therefore, this work is motivated by the memory challenge in the original progressive decoding [34] [35] and the results of [27].

Based on assessing the degree of Lagrange interpolation polynomials, we further propose a complexity reducing variant for the PASD-MM algorithm, namely the CR-PASD-MM algorithm. It can show the effectiveness of complexity reduction at high SNR. Complexity of the PASD-MM algorithm will be analyzed through characterizing the computational cost of image expansion and image reduction. It shows that the progressive MM interpolation yields a lower complexity for high rate codes, which is preferred in practice. Our analysis also reveals the channel dependent feature of the proposed algorithms. Numerical results show for the popular (255, 239) RS code, the PASD-MM algorithm and its variant yield a complexity reduction over the original PASD algorithm that employs Koetter's interpolation by sometimes two orders of magnitude. More importantly, this low complexity progressive interpolation is realized without any additional memory cost. Simulation results on decoding performance will also be provided, demonstrating that the proposed algorithms maintain the ASD error-correction capability.

The rest of the paper is organized as follows. Section II introduces RS codes and the PASD algorithm. Section III briefly reviews the known ASD-MM algorithm. Section IV introduces our proposed PASD-MM algorithm. Section V further introduces its complexity reducing variant. Section VI analyzes the complexity of the proposed algorithms. Section VII shows their decoding performance. Finally, Section VIII concludes the paper.

II. Background Knowledge

This section provides the background knowledge for the paper, including RS codes and the PASD algorithm [34].

A. RS Codes

Let $\mathbb{F}_q = \{\sigma_0, \sigma_1, \ldots, \sigma_{q-1}\}$ denote a finite field of size q, $\mathbb{F}_q[x]$ and $\mathbb{F}_q[x, y]$ denote the univariate and bivariate polynomial rings defined over $\mathbb{F}_q$, respectively. Given an (n, k) RS code with length n and dimension k, respectively, the message polynomial $f(x) \in \mathbb{F}_q[x]$ is defined as

$$f(x) = f_0 + f_1 x + \cdots + f_{k-1} x^{k-1},$$

where $f_0, f_1, \ldots, f_{k-1}$ are message symbols. Codeword $\underline{c} = (c_0, c_1, \ldots, c_{n-1}) \in \mathbb{F}_q^n$ can be generated by

$$\underline{c} = (f(\alpha_0), f(\alpha_1), \ldots, f(\alpha_{n-1})),$$

where $\alpha_0, \alpha_1, \ldots, \alpha_{n-1}$ are the n distinct nonzero elements of $\mathbb{F}_q$. They are called code locators.

B. The PASD Algorithm

Assume $\underline{c}$ is transmitted through a discrete memoryless channel and $\underline{\mathrm{r}} = (\mathrm{r}_0, \mathrm{r}_1, \ldots, \mathrm{r}_{n-1}) \in \mathbb{R}^n$ is the received vector. A reliability matrix $\mathbf{\Pi} \in \mathbb{R}_{\geq 0}^{q \times n}$ can be obtained based on $\underline{\mathrm{r}}$. Its entry π_{ij} is the symbol wise *a posteriori* probability [1] defined as $\pi_{ij} = \Pr[c_j = \sigma_i \mid \mathrm{r}_j]$, where $i = 0, 1, \ldots, q-1$ and $j = 0, 1, \ldots, n-1$. Matrix $\mathbf{\Pi}$ will be transformed into a multiplicity matrix $\mathbf{M} \in \mathbb{Z}_{\geq 0}^{q \times n}$. This step can be implemented by several approaches [9]–[11]. Its entry m_{ij} indicates the interpolation multiplicity for point (α_j, σ_i). Based on $\mathbf{M}$, interpolation finds the minimum polynomial $Q(x, y)$ that

[1] It is assumed that $\Pr[c_j = \sigma_i] = \frac{1}{q}, \forall (i, j)$.

interpolates all points (α_j, σ_i) with their prescribed multiplicity m_{ij}. Given a polynomial $Q \in \mathbb{F}_q[x, y]$ which can be written as $Q(x, y) = \sum_{a,b} Q_{ab} x^a y^b$ and a nonnegative integer pair (r, s), the (r, s)-Hasse derivative evaluation at point (α_j, σ_i) is defined as [36]

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

It implies an interpolation constraint on polynomial Q. If $D_{r,s}(Q(x, y))|_{(\alpha_j,\sigma_i)} = 0, \forall r + s < m_{ij}$, then Q interpolates point (α_j, σ_i) with a multiplicity of m_{ij}. Hence, there exists $\binom{m_{ij}+1}{2}$ interpolation constraints for point (α_j, σ_i) and matrix $\mathbf{M}$ contains $\sum_{i=0}^{q-1} \sum_{j=0}^{n-1} \binom{m_{ij}+1}{2}$ such constraints. For polynomial Q, its monomials $x^a y^b$ can be organized under the $(1, k-1)$-reverse lexicographical (revlex) order.[2] Let $x^{a'} y^{b'}$ denote the leading monomial of Q where $Q_{a'b'} \neq 0$, the $(1, k-1)$-weighted degree of Q is $\deg_{1,k-1} Q = \deg_{1,k-1} x^{a'} y^{b'}$. Furthermore, given two polynomials Q_1 and Q_2 with leading monomials $x^{a'_1} y^{b'_1}$ and $x^{a'_2} y^{b'_2}$, respectively, we declare $Q_1 < Q_2$ if $x^{a'_1} y^{b'_1} < x^{a'_2} y^{b'_2}$.

Definition 1 [9]: Given a multiplicity matrix $\mathbf{M}$ and a vector $\underline{\mu} = (\mu_0, \mu_1, \ldots, \mu_{n-1}) \in \mathbb{F}_q^n$, let $i_j = \text{index}\{\sigma_i \mid \sigma_i = \mu_j\}$, the matrix $\mathbf{M}$ based score of $\underline{\mu}$ is defined as

$$S_{\mathbf{M}}(\underline{\mu}) = \sum_{j=0}^{n-1} m_{i_j j}.$$

Theorem 1 [9]: Given an (n, k) RS code, let $Q(x, y) \in \mathbb{F}_q[x, y]$ denote an interpolation polynomial that satisfies the interpolation constraints defined by $\mathbf{M}$. If the score of codeword $\underline{c}$ satisfies $S_{\mathbf{M}}(\underline{c}) > \deg_{1,k-1} Q(x, y)$, then $Q(x, f(x)) = 0$.

Therefore, interpolation aims to find the polynomial Q that has the minimum $(1, k-1)$-weighted degree, and the message $f(x)$ can be retrieved by finding the y-roots of Q [8]. The maximum decoding output list size is determined by $\deg_y Q$. In this paper, we let $l = \deg_y Q$ which is the decoding parameter.

The interpolation is often implemented by Koetter's iterative polynomial construction algorithm [7]. It starts with initializing a set of $l + 1$ polynomials as $\mathcal{G}_l = \{1, y, \ldots, y^l\}$. They are iteratively updated to satisfy all the interpolation constraints defined by $\mathbf{M}$. The updated set is a Gröbner basis [19] in which Q is the minimum candidate. In contrast, the PASD algorithm functions with a progressively enlarged y-degree of the interpolation polynomial, which is denoted as v and $1 \leq v \leq l$. Based on $\mathbf{\Pi}$, a series of multiplicity matrices $\mathbf{M}_1, \mathbf{M}_2, \ldots, \mathbf{M}_l$ are generated accordingly. Beginning with $v = 1$, polynomial set $\mathcal{G}_1$ is initialized as $\{1, y\}$. Its entries will be computed to satisfy the interpolation constraints defined by $\mathbf{M}_1$. The interpolation polynomial Q_1 where $\deg_y Q_1 = 1$ is the minimum candidate of the computed set $\mathcal{G}_1$. If $Q_1(x, f(x)) = 0$, the message can be decoded [3] and the decoding terminates. Otherwise, $\mathcal{G}_1$ will be expanded by introducing a new polynomial y^2. The new polynomial needs to be updated to satisfy the interpolation constraints that have been satisfied by the existing polynomials of $\mathcal{G}_1$. The expanded polynomial set $\mathcal{G}_2$ will be further computed to satisfy the extra interpolation constraints defined by $\mathbf{M}_2$. As a result, the interpolation polynomial Q_2 where $\deg_y Q_2 = 2$ is the minimum candidate of the computed set $\mathcal{G}_2$. Again, if $Q_2(x, f(x)) = 0$, the decoding terminates. Otherwise, the decoding continues by enlarging v as above. It terminates either when the message is decoded or when v exceeds the predefined value l. Consequently, the PASD algorithm decodes the message with the smallest parameter v. The above description shows that the newly introduced polynomial needs to be updated using the intermediate decoding information. Hence, the progressive decoding system is realized with a certain memory requirement [34].

III. The ASD-MM Algorithm

This section reviews the ASD-MM algorithm which consists of $\mathbf{\Pi} \rightarrow \mathbf{M}$ transform [9], basis construction [22] [25], basis reduction [20] and root-finding [8]. They substantiate the proposed PASD-MM algorithm.

A. Basis Construction

In order to determine the interpolation polynomial $Q(x, y)$ where $\deg_y Q = l$, a module basis is needed. We first define module $\mathcal{M}_l$.

Definition 2: Given a multiplicity matrix $\mathbf{M}$, module $\mathcal{M}_l$ for the ASD is defined as the space of all bivariate polynomials over $\mathbb{F}_q[x, y]$ that interpolate all points (α_j, σ_i) with their multiplicity m_{ij} $(m_{ij} \neq 0)$. They have a maximum y-degree of l.

In this paper, we utilize Algorithm A of [9] to perform the $\mathbf{\Pi} \rightarrow \mathbf{M}$ transform. Let

$$\mathsf{m}_j = \sum_{i=0}^{q-1} m_{ij}$$

and

$$\mathsf{m} = \max\{\mathsf{m}_j, \forall j\}.$$

The transform stops when m reaches the predefined l. Therefore, $\mathsf{m}_j \leq l$ and $\mathsf{m} = l$. As l becomes infinity, the multiplicity matrix $\mathbf{M}$ would be proportional to the reliability matrix $\mathbf{\Pi}$ [9]. In general, a larger decoding parameter l leads to a better decoding capability. Note that several other techniques [10] [11] can be applied for the $\mathbf{\Pi} \rightarrow \mathbf{M}$ transform to improve the ASD performance, but with a higher transform complexity.

Now we construct the basis of module $\mathcal{M}_l$. It can be underpinned by the following point enumeration [22]. Let L_j denote an enumeration list that is drawn from column j of $\mathbf{M}$.

[2] The $(1, k-1)$-weighted degree of $x^a y^b$ is $\deg_{1,k-1} x^a y^b = a + (k-1)b$. Given two distinct monomials $x^{a_1} y^{b_1}$ and $x^{a_2} y^{b_2}$, $x^{a_1} y^{b_1} < x^{a_2} y^{b_2}$ 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 $b_1 < b_2$.

[3] The message polynomial $f(x)$ can be validated using the maximum likelihood (ML) criterion of [37].

It contains interpolation points (α_j, σ_i) with their multiplicity m_{ij} as

$$L_j = [\underbrace{(\alpha_j, \sigma_i), \ldots, (\alpha_j, \sigma_i)}_{m_{ij}}, \forall i \text{ and } m_{ij} \neq 0].$$

Note that $|L_j| = \mathsf{m}_j$. Its balanced list L'_j is further created as follows. Initialize $L'_j = \emptyset$. Move one of the most frequent elements from L_j to the back of L'_j and repeat this process m_j times until L_j becomes empty. The balanced list can be denoted as

$$L'_j = [(\alpha_j, y_j^{(0)}), (\alpha_j, y_j^{(1)}), \ldots, (\alpha_j, y_j^{(\mathsf{m}_j - 1)})], \tag{1}$$

where $y_j^{(0)}, y_j^{(1)}, \ldots, y_j^{(\mathsf{m}_j-1)} \in \mathbb{F}_q$ and they appear with the exact multiplicities given by m_{ij}. Since L'_j is a permutation of L_j, $|L'_j| = \mathsf{m}_j$. Finally, let $\mathsf{m}_j(t)$ denote the maximum multiplicity of the last $\mathsf{m}_j - t$ elements of L'_j as

$$\mathsf{m}_j(t) = \max\{\text{multi.}((\alpha_j, y_j^{(t)})), \ldots, \text{multi.}((\alpha_j, y_j^{(\mathsf{m}_j-1)}))\}.$$

Note that $\mathsf{m}_j(0) = \max\{m_{ij}, \forall i\}$ and $\mathsf{m}_j(t) = 0$ for $t \geq \mathsf{m}_j$.

Module $\mathcal{M}_l$ can now be generated. First, we define the following *Lagrange interpolation polynomials*

$$F_\varepsilon(x) = \sum_{j \in \Upsilon_\varepsilon} y_j^{(\varepsilon)} \prod_{j' \in \Upsilon_\varepsilon, j' \neq j} \frac{x - \alpha_{j'}}{\alpha_j - \alpha_{j'}}, \tag{2}$$

where $\Upsilon_\varepsilon = \{j \mid \mathsf{m}_j(\varepsilon) > 0\}$ and $\varepsilon = 0, 1, \ldots, l-1$. It holds that $F_\varepsilon(\alpha_j) = y_j^{(\varepsilon)}, \forall j \in \Upsilon_\varepsilon$. Therefore, $y - F_\varepsilon(x)$ interpolates points $(\alpha_j, y_j^{(\varepsilon)}), \forall j \in \Upsilon_\varepsilon$. Now, $\mathcal{M}_l$ can be generated as an $\mathbb{F}_q[x]$-module by the following $l+1$ polynomials [22] [25]

$$P_t(x, y) = \prod_{j=0}^{n-1} (x - \alpha_j)^{\mathsf{m}_j(t)} \prod_{\varepsilon=0}^{t-1} (y - F_\varepsilon(x)), \tag{3}$$

where $t = 0, 1, \ldots, l$.

Lemma 2 [22]: Let $\mathcal{Q}_t(x, y) = \sum_{\tau=0}^{t} \mathcal{Q}_t^{(\tau)}(x) y^\tau \in \mathcal{M}_l$ with $\deg_y \mathcal{Q}_t = t < l$, we have $\prod_{j=0}^{n-1}(x - \alpha_j)^{\mathsf{m}_j(t)} | \mathcal{Q}_t^{(t)}(x)$.

Consequently, we prove the following Theorem.

Theorem 3: Polynomials $P_t(x, y)$ form a basis of $\mathcal{M}_l$.

Proof: First, we prove $P_t(x, y) \in \mathcal{M}_l$. It can be seen that $\prod_{\varepsilon=0}^{t-1}(y - F_\varepsilon(x))$ interpolates the first t points of all balanced lists while $\prod_{j=0}^{n-1}(x - \alpha_j)^{\mathsf{m}_j(t)}$ interpolates the remaining points. Since $\deg_y P_t(x, y) \leq l, \forall t$, recalling Definition 2, $P_t(x, y) \in \mathcal{M}_l$.

Next, we prove any element of $\mathcal{M}_l$ can be presented as an $\mathbb{F}_q[x]$-linear combination of $P_t(x, y)$. Assume that $\mathcal{Q}(x, y) \in \mathcal{M}_l$ and let us write (3) as $P_t(x, y) = \sum_{\tau=0}^{t} P_t^{(\tau)}(x) y^\tau$. Since when $t = l$, $P_l^{(l)}(x) = 1$, there exists a polynomial $p_l(x) \in \mathbb{F}_q[x]$ that enables $\mathcal{Q}_{l-1}(x, y) = \mathcal{Q}(x, y) - p_l(x) P_l(x, y)$ so that $\deg_y \mathcal{Q}_{l-1} = l - 1$. Note that if $\deg_y \mathcal{Q} < l$, $p_l(x) = 0$. Since $\mathcal{Q}, P_l \in \mathcal{M}_l$, then $\mathcal{Q}_{l-1} \in \mathcal{M}_l$. Continuing with $t = l-1$, $P_{l-1}^{(l-1)}(x) = \prod_{j=0}^{n-1}(x - \alpha_j)^{\mathsf{m}_j(l-1)}$. Based on Lemma 2, $\prod_{j=0}^{n-1}(x - \alpha_j)^{\mathsf{m}_j(l-1)} | \mathcal{Q}_{l-1}^{(l-1)}(x)$. Therefore, we can generate $\mathcal{Q}_{l-2}(x, y)$ by $\mathcal{Q}_{l-2}(x, y) = \mathcal{Q}_{l-1}(x, y) - p_{l-1}(x) P_{l-1}(x, y)$ so that $\deg_y \mathcal{Q}_{l-2} = l - 2$. Following the above deduction until $t = 0$, we have $P_0^{(0)}(x) = \prod_{j=0}^{n-1}(x - \alpha_j)^{\mathsf{m}_j(0)}$ and $\prod_{j=0}^{n-1}(x - \alpha_j)^{\mathsf{m}_j(0)} | \mathcal{Q}_0^{(0)}(x)$. Hence, there exists $p_0(x)$ that enables $\mathcal{Q}_0(x, y) - p_0(x) P_0(x, y) = 0$. Therefore, if $\mathcal{Q} \in \mathcal{M}_l$, it can be written as an $\mathbb{F}_q[x]$-linear combination of $P_t(x, y)$, i.e., $\mathcal{Q}(x, y) = \sum_{t=0}^{l} p_t(x) P_t(x, y)$.

Therefore, equation (3) defines a basis of $\mathcal{M}_l$, denoted as $\mathcal{B}_l$. ■

The following Remark points out the ASD-MM algorithm can be simplified into a hard-decision decoding, i.e., the MM based GS algorithm [18].

Remark 4: Let $\underline{\omega} = (\omega_0, \omega_1, \ldots, \omega_{n-1})$ denote the hard-decision received word. For the GS algorithm, interpolation determines polynomial $Q(x, y)$ that interpolates the n points (α_0, ω_0), $(\alpha_1, \omega_1), \ldots, (\alpha_{n-1}, \omega_{n-1})$ with a multiplicity of m $(m \leq l)$ [6]. This implies that $y_j^{(\varepsilon)} = \omega_j$, where $\varepsilon = 0, 1, \ldots, m-1$. Therefore, $\mathsf{m}_j(t) = m - t$ and $|\Upsilon_t| = n$ for $t = 0, 1, \ldots, m-1$, $\mathsf{m}_j(t) = 0$ and $|\Upsilon_t| = \emptyset$ for $t = m, m+1, \ldots, l-1$. The Lagrange interpolation polynomial is simplified to

$$F(x) = \sum_{j=0}^{n-1} \omega_j \prod_{j'=0, j' \neq j}^{n-1} \frac{x - \alpha_{j'}}{\alpha_j - \alpha_{j'}}$$

and the module generators of (3) become

$$P_t(x, y) = \prod_{j=0}^{n-1} (x - \alpha_j)^{m-t} (y - F(x))^t, \quad \text{if } t = 0, 1, \ldots, m-1,$$

$$P_t(x, y) = y^{t-m} (y - F(x))^m, \quad \text{if } t = m, m+1, \ldots, l.$$

B. Basis Reduction

In order to describe the basis reduction, we need to present $\mathcal{B}_l$ as a matrix over $\mathbb{F}_q[x]$.

Definition 3: Given a matrix $\mathcal{V} \in \mathbb{F}_q[x]^{(l+1) \times (l+1)}$, let t and τ denote its row index and column index, respectively. Further let $\mathcal{V}|_t$ denote its row-t and $\mathcal{V}|_t^{(\tau)}$ denote its entry of row-t column-τ.

- The row-degree of $\mathcal{V}|_t$ is $\text{rdeg}\, \mathcal{V}|_t = \max\{\deg \mathcal{V}|_t^{(\tau)}, \forall \tau\}$.
- The leading position (LP) of $\mathcal{V}|_t$ is $\text{LP}(\mathcal{V}|_t) = \max\{\tau \mid \deg \mathcal{V}|_t^{(\tau)} = \text{rdeg}\, \mathcal{V}|_t\}$.
- The degree of matrix $\mathcal{V}$ is $\text{mdeg}\, \mathcal{V} = \sum_t \text{rdeg}\, \mathcal{V}|_t$.

Let $[\![\mathbb{F}_q[x, y]]\!]_l = \{Q \in \mathbb{F}_q[x, y] \mid \deg_y Q \leq l\}$, we define a bijective map from a bivariate polynomial $Q(x, y) = \sum_{\tau \leq l} Q^{(\tau)}(x) y^\tau$ to a vector over $\mathbb{F}_q[x]$ as

$$\phi_l : [\![\mathbb{F}_q[x, y]]\!]_l \rightarrow \mathbb{F}_q[x]^{l+1}$$
$$Q^{(0)}(x) + \cdots + Q^{(l)}(x) y^l \mapsto (Q^{(0)}(x), \ldots, Q^{(l)}(x)).$$

Now we can present basis $\mathcal{B}_l$ as a matrix over $\mathbb{F}_q[x]$ by letting $\mathcal{B}_l|_t = \phi_l(P_t(x, y)), \forall t$. $\mathcal{B}_l$ will be further reduced into the Gröbner basis [19] of $\mathcal{M}_l$. The following Proposition gives a simple criterion for validating the Gröbner basis.

Proposition 5 [25]: Assume that $\{g_t \in [\![\mathbb{F}_q[x, y]]\!]_l, 0 \leq t \leq l\}$ generates module $\mathcal{M}_l$. Under the $(1, k-1)$-revlex order, if y-degree of the leading monomial of each polynomial g_t is different, $\{g_t \in [\![\mathbb{F}_q[x, y]]\!]_l, 0 \leq t \leq l\}$ is a Gröbner basis of $\mathcal{M}_l$.

In this paper, we utilize the MS algorithm [20] to realize the basis reduction. Several research [21]–[24] had proposed the asymptotically faster algorithms which employ fast polynomial multiplication. But they are also markedly more involved to implement. It is outside the scope of this paper to implement and see for which codeword length they become faster than the MS algorithm.

Definition 4 [20]: Given a square matrix $\mathcal{V}$ over $\mathbb{F}_q[x]$, if any two rows $\mathcal{V}|_t$ and $\mathcal{V}|_{t'}$ exhibit $\text{LP}(\mathcal{V}|_t) \neq \text{LP}(\mathcal{V}|_{t'})$, then $\mathcal{V}$ is in *weak Popov form*.

Lemma 6 [38]: For a square matrix $\mathcal{V}$ over $\mathbb{F}_q[x]$, when it is in weak Popov form, we have $\text{mdeg}\,\mathcal{V} = \deg \det \mathcal{V}$.

After basis $\mathcal{B}_l$ is constructed by (3), it will be mapped by

$$\mathcal{A}_l = \mathcal{B}_l \cdot \text{diag}(1, x^{k-1}, \ldots, x^{l(k-1)}),$$

so that $\text{rdeg}\,\mathcal{A}_l|_t = \deg_{1,k-1} P_t(x,y)$. The MS algorithm [20] will reduce $\mathcal{A}_l$ into weak Popov form $\mathcal{A}'_l$. Demap it by

$$\mathcal{B}'_l = \mathcal{A}'_l \cdot \text{diag}(1, x^{-(k-1)}, \ldots, x^{-l(k-1)}),$$

and let $P'_t(x,y) = \phi_l^{-1}(\mathcal{B}'_l|_t), \forall t$. Since $\deg_{1,k-1} P'_t(x,y) = \text{rdeg}\,\mathcal{A}'_l|_t = \deg \mathcal{A}'_l|_t^{(\text{LP}(\mathcal{A}'_l|_t))}$, when $\mathcal{A}'_l$ is in weak Popov form, y-degree of each polynomial's leading monomial, i.e., $\text{LP}(\mathcal{A}'_l|_t)$, is different. Based on Proposition 5, $\mathcal{B}'_l$ is the Gröbner basis of $\mathcal{M}_l$ under the $(1, k-1)$-revlex order. Among the polynomials $P'_t(x,y)$, the minimum one (also under the same order) is chosen as the interpolation polynomial $Q(x,y)$. Root-finding further determines its y-roots [8]. If multiple y-roots are found, output the estimated message whose corresponding codeword has the minimum Euclidean distance to the received vector $\underline{\text{r}}$.

IV. The PASD-MM Algorithm

This section introduces the proposed PASD-MM algorithm. It progressively increases y-degree of the interpolation polynomial, i.e., the decoding output list size, leading to a gradually enhanced error-correction capability. Again, v denotes the progressive iteration index and $1 \leq v \leq l$. We first introduce the concepts of submodule and its basis image.

A. Submodule and Its Basis Image

We introduce submodule that is the subspace of a module. It is defined as follows.

Definition 5: Given a module $\mathcal{M}_l$ that is generated by (3), its submodule $\mathfrak{M}_v$ is the subspace spanned by $P_0(x,y), \ldots, P_v(x,y)$.

Therefore, $P_0(x,y), \ldots, P_v(x,y)$ form a basis $\mathfrak{B}_v$ of $\mathfrak{M}_v$. Note that $\deg_y P_t(x,y) \leq v$ where $t = 0, 1, \ldots, v$, and $\mathfrak{B}_v \in \mathbb{F}_q[x]^{(v+1)\times(v+1)}$.

For a balanced list L'_j, we define

$$\delta_j(t) = \mathsf{m}_j(t) - \mathsf{m}_j(t+1), \tag{4}$$

where $t = 0, 1, \ldots, l$. Since $\mathsf{m}_j \leq l$, $\mathsf{m}_j(l+1) = \mathsf{m}_j(l) = 0$. Consequently, $\delta_j(l-1) = \mathsf{m}_j(l-1)$ and $\delta_j(l) = 0$. Let us further define

$$G_t(x) = \prod_{j=0}^{n-1} (x - \alpha_j)^{\mathsf{m}_j(t)} \tag{5}$$

and

$$R_t(x) = \prod_{j=0}^{n-1} (x - \alpha_j)^{\delta_j(t)}. \tag{6}$$

Based on (4), it can be realized that

$$\begin{aligned} G_t(x) &= G_{t+1}(x) R_t(x) \\ &= G_v(x) \prod_{\epsilon=t}^{v-1} R_\epsilon(x), \end{aligned} \tag{7}$$

where $v = t+1, t+2, \ldots, l$. Since $\mathsf{m}_j(l) = \delta_j(l) = 0, \forall j$, $G_l(x) = R_l(x) = 1$.

Let $\Theta_t^\tau = \{\theta \subset \{0, 1, \ldots, t-1\} \mid |\theta| = \tau\}$. Note that $\Theta_t^0 = \{\emptyset\}$, $|\Theta_t^\tau| = \binom{t}{\tau}$ and $|\Theta_t^t| = 1$. For example, $\Theta_4^2 = \{\{0,1\}, \{0,2\}, \{0,3\}, \{1,2\}, \{1,3\}, \{2,3\}\}$ and $|\Theta_4^2| = 6$.

With the above notations, generators (3) can be rewritten as

$$P_t(x,y) = G_t(x) W_t(x,y), \tag{8}$$

where

$$W_t(x,y) = \prod_{\varepsilon=0}^{t-1} (y - F_\varepsilon(x)) = \sum_{\tau=0}^{t} w_t^{(\tau)}(x) y^\tau \tag{9}$$

and

$$w_t^{(\tau)}(x) = \sum_{\theta \in \Theta_t^{t-\tau}} \prod_{\varepsilon \in \theta} (-F_\varepsilon(x)). \tag{10}$$

Note that $w_t^{(t)}(x) = 1$ and $W_0(x,y) = 1$. The following Theorem characterizes the recursive structure of $\mathfrak{B}_v$, which underpins the PASD-MM algorithm.

Theorem 7: Let $\Xi_0 = [1]$, basis $\mathfrak{B}_v$ can be written as

$$\mathfrak{B}_v = G_v(x) \cdot \Xi_v, \tag{11}$$

where

$$\Xi_v = \begin{bmatrix} \multicolumn{3}{c}{R_{v-1}(x) \cdot \Xi_{v-1}} & \mathbf{0}_v^{\mathrm{T}} \\ w_v^{(0)}(x) & \cdots & w_v^{(v-1)}(x) & w_v^{(v)}(x) \end{bmatrix} \tag{12}$$

and $1 \leq v \leq l$. Note that $\mathbf{0}_v$ denotes an all zero vector of size v.

Proof: Based on (8), when $v = 1$, $\mathfrak{B}_1$ contains

$$\begin{aligned} P_0(x,y) &= G_0(x) W_0(x,y), \\ P_1(x,y) &= G_1(x) W_1(x,y). \end{aligned}$$

Since $G_0(x) = G_1(x) R_0(x)$ and $W_0(x,y) = 1$, $\mathfrak{B}_1 = G_1(x) \cdot \Xi_1$, where

$$\Xi_1 = \begin{bmatrix} R_0(x) \cdot \Xi_0 & 0 \\ w_1^{(0)}(x) & w_1^{(1)}(x) \end{bmatrix}.$$

Note that $\mathcal{P}_{1,0}(x,y) = \phi_1^{-1}(\Xi_1|_0) = R_0(x)$ and $\mathcal{P}_{1,1}(x,y) = \phi_1^{-1}(\Xi_1|_1) = W_1(x,y)$.

Based on (7) and (8), we know when $v \geq 2$, $\mathfrak{B}_{v-1}$ contains

$$\begin{aligned} P_t(x,y) &= G_{v-1}(x) \prod_{\epsilon=t}^{v-2} R_\epsilon(x) W_t(x,y), \text{if } t = 0, 1, \ldots, v-2, \\ P_{v-1}(x,y) &= G_{v-1}(x) W_{v-1}(x,y). \end{aligned}$$

Therefore, $\mathfrak{B}_{v-1} = G_{v-1}(x) \cdot \Xi_{v-1}$, and $\mathcal{P}_{v-1,t}(x,y) = \phi_{v-1}^{-1}(\Xi_{v-1}|_t) = \prod_{\epsilon=t}^{v-2} R_\epsilon(x) W_t(x,y)$ for $t = 0, 1, \ldots, v-1$ and $\mathcal{P}_{v-1,v-1}(x,y) = W_{v-1}(x,y)$. Furthermore, $\mathfrak{B}_v$ contains

$$P_t(x,y) = G_v(x) \prod_{\epsilon=t}^{v-1} R_\epsilon(x) W_t(x,y), \text{ if } t = 0, 1, \ldots, v-1,$$
$$P_v(x,y) = G_v(x) W_v(x,y).$$

Therefore,

$$\mathfrak{B}_v = G_v(x) \cdot \Xi_v,$$

and $\mathcal{P}_{v,t}(x,y) = \phi_v^{-1}(\Xi_v|_t) = \prod_{\epsilon=t}^{v-1} R_\epsilon(x) W_t(x,y) = R_{v-1}(x)\mathcal{P}_{v-1,t}(x,y) = R_{v-1}(x)\phi_{v-1}^{-1}(\Xi_{v-1}|_t)$ for $t = 0, 1, \ldots, v-1$ and $\mathcal{P}_{v,v}(x,y) = \phi_v^{-1}(\Xi_v|_v) = W_v(x,y)$. Consequently, $\Xi_v|_t = R_{v-1}(x) \cdot \Xi_{v-1}|_t$ for $t = 0, 1, \ldots, v-1$, and $\Xi_v|_v = \phi_v(W_v(x,y))$. The recursive structure of (12) can be obtained. ■

Definition 6: Define the following bijective map

$$\psi_v : \mathbb{F}_q[x]^{(v+1)\times(v+1)} \to \mathbb{F}_q[x]^{(v+1)\times(v+1)}$$
$$[\mathfrak{B}_v|_t^{(\tau)}, \forall(t,\tau)] \mapsto \Big[\frac{\mathfrak{B}_v|_t^{(\tau)}}{G_v(x)}, \forall(t,\tau)\Big],$$

where $(t,\tau) = 0, 1, \ldots, v$. Under such map, $\Xi_v = \psi_v(\mathfrak{B}_v)$ and it is called the image of submodule basis $\mathfrak{B}_v$.

Based on Theorem 7 and since $G_l(x) = 1$, we have $\mathfrak{B}_l = \Xi_l$. Further recalling Definition 5, we have $\mathfrak{B}_l = \mathcal{B}_l$. Therefore, the following Corollary can be led to.

Corollary 8: When reaching the last progressive iteration as $v = l$, the submodule basis and its image are equivalent to module basis $\mathcal{B}_l$ as $\mathfrak{B}_l = \Xi_l = \mathcal{B}_l$.

Let us denote two diagonal matrices as

$$\mathcal{D}_v = \text{diag}(1, x^{k-1}, \ldots, x^{v(k-1)})$$

and

$$\mathcal{D}_v^{-1} = \text{diag}(1, x^{-(k-1)}, \ldots, x^{-v(k-1)}).$$

Theorem 7 and Corollary 8 reveal that $\mathcal{B}_l$ can be progressively constructed through the images of its submodule basis. The MS algorithm performs $\mathbb{F}_q[x]$-linear combinations for rows of $\mathcal{A}_l$. This row operation can be rescheduled as the following. The MS algorithm can target the first two rows of $\mathcal{A}_l$. This is equivalent to reducing matrix $\mathfrak{B}_1 \cdot \mathcal{D}_1$ into weak Popov form. It then targets the first three rows of $\mathcal{A}_l$, which is equivalent to reducing matrix $\mathfrak{B}_2 \cdot \mathcal{D}_2$ into weak Popov form. Continue the process until matrix $\mathfrak{B}_l \cdot \mathcal{D}_l$ is in weak Popov form. Since $G_v(x)$ is the common multiplier of all polynomials of $\mathfrak{B}_v$, performing the MS algorithm on matrix $\mathfrak{B}_v \cdot \mathcal{D}_v$ is equivalent to performing it on matrix $\Xi_v \cdot \mathcal{D}_v$. This leads to the following PASD-MM algorithm. It aims to decode the message from an intermediate interpolation polynomial $Q_v(x,y)$ where $\deg_y Q_v = v$, which will be retrieved from the reduced matrix $\Xi_v \cdot \mathcal{D}_v$.

B. The Algorithm

The PASD-MM algorithm decodes the message from the image of the progressively enlarged submodule basis. The progressive interpolation consists of two steps, image expansion and image reduction. At the beginning, $v = 1$, image Ξ_1 is initialized as

$$\mathcal{P}_{1,0}(x,y) = R_0(x),$$
$$\mathcal{P}_{1,1}(x,y) = W_1(x,y).$$

Map Ξ_1 into $\mathcal{X}_1 = \Xi_1 \cdot \mathcal{D}_1$ and the MS algorithm will reduce $\mathcal{X}_1$ into weak Popov form $\mathcal{X}_1'$. Demap it as $\Xi_1' = \mathcal{X}_1' \cdot \mathcal{D}_1^{-1}$. Polynomials $\mathcal{P}_{1,0}'(x,y)$ and $\mathcal{P}_{1,1}'(x,y)$ can be retrieved from Ξ_1' by $\mathcal{P}_{1,0}'(x,y) = \phi_1^{-1}(\Xi_1'|_0)$ and $\mathcal{P}_{1,1}'(x,y) = \phi_1^{-1}(\Xi_1'|_1)$, respectively. Among them, the minimum one is chosen as the interpolation polynomial $Q_1(x,y)$ where $\deg_y Q_1 = 1$. Further determine y-root of Q_1. If $Q_1(x, \hat{f}(x)) = 0$ and the estimated codeword $\underline{\hat{c}} = (\hat{f}(\alpha_0), \hat{f}(\alpha_1), \ldots, \hat{f}(\alpha_{n-1}))$ satisfies the ML criterion [37] (refer to Appendix A), the decoding terminates and outputs $\hat{f}(x)$. Otherwise, the decoding progresses to determine $Q_2(x,y)$ through expanding Ξ_1' to Ξ_2.

In general, at progressive iteration $v-1$ ($v \geq 2$), if the message cannot be decoded from Ξ_{v-1}', then Ξ_{v-1}' will be expanded to Ξ_v in order to find $Q_v(x,y)$. Based on Theorem 7, Ξ_v can be generated by

$$\mathcal{P}_{v,t}(x,y) = R_{v-1}(x)\mathcal{P}_{v-1,t}'(x,y), \text{ if } t = 0, 1, \ldots, v-1, \tag{13}$$

$$\mathcal{P}_{v,v}(x,y) = W_v(x,y), \tag{14}$$

where $\mathcal{P}_{v-1,t}'(x,y) = \phi_{v-1}^{-1}(\Xi_{v-1}'|_t)$. Based on (9) and (10), we know $\mathcal{P}_{v,v}(x,y)$ can be directly generated based on the balanced lists, which does not require the knowledge of the intermediate decoding information. This overcomes the memory cost of the original PASD algorithm [34]. After generating Ξ_v, it will be mapped by

$$\mathcal{X}_v = \Xi_v \cdot \mathcal{D}_v. \tag{15}$$

The MS algorithm will then reduce $\mathcal{X}_v$ into weak Popov form $\mathcal{X}_v'$. Further demap it as

$$\Xi_v' = \mathcal{X}_v' \cdot \mathcal{D}_v^{-1}. \tag{16}$$

Polynomials $\mathcal{P}_{v,0}'(x,y), \ldots, \mathcal{P}_{v,v}'(x,y)$ are retrieved from Ξ_v' by $\mathcal{P}_{v,0}'(x,y) = \phi_v^{-1}(\Xi_v'|_0), \ldots, \mathcal{P}_{v,v}'(x,y) = \phi_v^{-1}(\Xi_v'|_v)$, respectively. Among them, the minimum one is chosen as $Q_v(x,y)$. If $Q_v(x, \hat{f}(x)) = 0$ and the estimated codeword $\underline{\hat{c}}$ satisfies the ML criterion, the decoding terminates and output $\hat{f}(x)$. Note that Q_v may have multiple y-roots, but only one of them would satisfy the criterion. If the ML codeword is not found, the decoding progresses by updating $v = v + 1$. If $v > l$, it implies the designed maximum y-degree of the interpolation polynomial is exceeded. The decoding terminates with a decoding failure, i.e., no ML codeword is found. Otherwise, the decoding continues.

The PASD-MM algorithm is summarized in Algorithm 1.

C. Validity Analysis

Let $\tilde{\mathbf{M}}_v \in \mathbb{Z}_{\geq 0}^{q\times n}$ denote a multiplicity matrix where its entry $\tilde{m}_{ij}(v)$ defines the interpolation multiplicity that has been held by polynomials $\mathcal{P}_{v,0}'(x,y), \ldots, \mathcal{P}_{v,v}'(x,y)$ w.r.t. point (α_j, σ_i). Note that polynomials $\mathcal{P}_{v,t}'(x,y) = \phi_v^{-1}(\Xi_v'|_t)$.

Algorithm 1 The PASD-MM Algorithm

```
Input: $\mathbf{M}$;
Output: $\hat{f}(x)$;
1: Generate all balanced lists $L'_j$ as in (1);
2: Initialize $v = 1$ and $\mathcal{P}'_{0,0}(x, y) = 1$;
3: Generate $\Xi_v$ as in (13) and (14);
4: Map $\Xi_v$ to $\mathcal{X}_v$ as in (15);
5: Perform the MS algorithm to yield $\mathcal{X}'_v$;
6: Demap $\mathcal{X}'_v$ to $\Xi'_v$ as in (16) and determine $Q_v(x, y)$;
7: Determine $y$-roots of $Q_v$. If $Q_v(x, \hat{f}(x)) = 0$ and $\hat{\underline{c}}$
   satisfies the ML criterion, output $\hat{f}(x)$ and terminate the
   decoding; Otherwise, update $v = v + 1$;
8: If $v > l$, terminate the decoding and declare a failure;
   Otherwise, go to Step 3.
```

Since $\mathfrak{B}_v$ contains polynomials that interpolate the points with a multiplicity of at least m_{ij}, based on (11),

$$\tilde{m}_{ij}(v) \geq \max\{m_{ij} - \mathsf{m}_j(v), 0\}.$$

Let $\mathcal{Q}_v(x, y)$ and $Q_v(x, y)$ denote the minimum candidate of the reduced $\mathfrak{B}_v$ and Ξ_v, respectively. Polynomial $\mathcal{Q}_v(x, y)$ interpolates points (α_j, σ_i) with their multiplicity m_{ij} and its codeword score is $S_{\mathbf{M}}(\underline{c})$, while polynomial $Q_v(x, y)$ interpolates points (α_j, σ_i) with their reduced multiplicity $\tilde{m}_{ij}(v)$. Based on Definition 1, the $\tilde{\mathbf{M}}_v$ based score of $\underline{c}$ is

$$S_{\tilde{\mathbf{M}}_v}(\underline{c}) = \sum_{j=0}^{n-1} \tilde{m}_{i_j j}(v).$$

The following Theorem validates the process of finding message $f(x)$ from the progressively enlarged Ξ_v.

Theorem 9: Given an (n, k) RS code, if the transmitted codeword $\underline{c}$ satisfies $S_{\mathbf{M}}(\underline{c}) > \deg_{1,k-1} \mathcal{Q}_v(x, y)$, then $S_{\tilde{\mathbf{M}}_v}(\underline{c}) > \deg_{1,k-1} Q_v(x, y)$ and $Q_v(x, f(x)) = 0$.

Proof: Based on Theorem 7, we know $\mathcal{Q}_v(x, y) = G_v(x) \cdot Q_v(x, y)$, and

$$\deg_{1,k-1} \mathcal{Q}_v(x, y) = \deg G_v(x) + \deg_{1,k-1} Q_v(x, y).$$

Based on (5), we know $G_v(x)$ interpolates points (α_j, c_j) with a multiplicity of $\mathsf{m}_j(v)$. Therefore,

$$S_{\mathbf{M}}(\underline{c}) = \deg G_v(x) + S_{\tilde{\mathbf{M}}_v}(\underline{c}).$$

If $S_{\mathbf{M}}(\underline{c}) > \deg_{1,k-1} \mathcal{Q}_v(x, y)$, then $S_{\tilde{\mathbf{M}}_v}(\underline{c}) > \deg_{1,k-1} Q_v(x, y)$. Based on Theorem 1, we have $Q_v(x, f(x)) = 0$. ■

Therefore, retrieving $f(x)$ from $\mathfrak{B}_v$ is equivalent to retrieving it from Ξ_v. When $v = l$, $\tilde{\mathbf{M}}_l = \mathbf{M}$. This reveals that as l approaches infinity, $\tilde{\mathbf{M}}_l$ also becomes proportional to matrix $\mathbf{\Pi}$. The PASD-MM algorithm would maintain the optimal ASD performance. However, it should be pointed out that the intermediate $\tilde{\mathbf{M}}_v$ may not be proportional to $\mathbf{\Pi}$. Matrix $\tilde{\mathbf{M}}_v$ is obtained based on the recursive structure of (12), but not on the consideration of maximizing the successful intermediate decoding probability. The construction of $\tilde{\mathbf{M}}_v$ is related to the choice of interpolation points during the intermediate decoding. A more accurate assumption for the interpolation point distribution such as [10] [11] may help improve the intermediate decoding performance. This will be considered in our future work.

V. A Complexity Reduction Approach

We further propose a complexity reduction approach for the PASD-MM algorithm, naming it the CR-PASD-MM algorithm. It is based on assessing the degree of Lagrange interpolation polynomials $F_\varepsilon(x)$ of (2).

Lemma 10: Given $F_\varepsilon(x)$, if $|\Upsilon_\varepsilon| = n$ and $\underline{y}^{(\varepsilon)} = (y_0^{(\varepsilon)}, y_1^{(\varepsilon)}, \ldots, y_{n-1}^{(\varepsilon)})$ is a codeword, then $\deg F_\varepsilon(x) < k$.

Proof: Based on (2), we have $\deg F_\varepsilon(x) \leq n - 1$ and $F_\varepsilon(\alpha_j) = y_j^{(\varepsilon)}, \forall j \in \Upsilon_\varepsilon$. If $|\Upsilon_\varepsilon| = n$ and $\underline{y}^{(\varepsilon)}$ is a codeword, there exists a message polynomial $g(x) \in \mathbb{F}_q[x]$ with $\deg g(x) < k$ such that $g(\alpha_j) = y_j^{(\varepsilon)}, \forall j$.

Let $g'(x) = F_\varepsilon(x) - g(x)$, we have $\deg g'(x) \leq n-1$. Since $g'(\alpha_j) = F_\varepsilon(\alpha_j) - g(\alpha_j) = 0$, $g'(x)$ has n roots. It can be written as $g'(x) = \gamma(x) \cdot \prod_{j=0}^{n-1}(x - \alpha_j)$, where $\gamma(x) \in \mathbb{F}_q[x]$. This leads to $\deg g'(x) \geq n$, which contradicts to the fact that $\deg g'(x) \leq n-1$. Therefore, $\gamma(x) = 0$ and $g'(x) = 0$. Hence, $F_\varepsilon(x) = g(x)$ and $\deg F_\varepsilon(x) < k$. ■

Lemma 10 implies that during the progressive decoding, we can determine whether $\underline{y}^{(\varepsilon)}$ is a valid codeword by assessing the degree of $F_\varepsilon(x)$. If $\deg F_\varepsilon(x) < k$ and $\underline{y}^{(\varepsilon)}$ also satisfies the ML criterion [37], the decoding outputs $F_\varepsilon(x)$ as $\hat{f}(x)$ and terminates without performing image expansion and reduction of the current iteration. This reduces complexity of the PASD-MM algorithm. The CR-PASD-MM algorithm is further summarized as follows.

Algorithm 2 The CR-PASD-MM Algorithm

```
Input: $\mathbf{M}$;
Output: $\hat{f}(x)$;
1: Initialize $v = 1$;
2: Construct $F_{v-1}(x)$ as in (2);
3: If $\deg F_{v-1}(x) < k$ and $\underline{y}^{(v-1)}$ is an ML codeword
4:     Terminate the decoding and output $F_{v-1}(x)$ as $\hat{f}(x)$;
5: Else
6:     Perform Step 3 − 7 of Algorithm 1;
7: If $v > l$, terminate the decoding and declare a failure;
   Otherwise, go to Step 2.
```

The following Lemma can be further led to based on Lemma 10.

Lemma 11: At the progressive iteration v, if there exists a Lagrange interpolation polynomial $F_\varepsilon(x)$ with $\deg F_\varepsilon(x) > k - 1$ and $\varepsilon < v$, then the newly formulated matrix $\mathcal{X}_v$ will not be in weak Popov form.

Proof: At the progressive iteration v, since $w_v^{(v)}(x) = 1$, $\deg \mathcal{X}_v|_v^{(v)} = v(k-1)$. Without loss of generality, we assume there exists $\deg F_{\varepsilon'}(x) > k - 1$ and $\deg F_\varepsilon(x) \leq k - 1, \forall \varepsilon \neq \varepsilon'$. Since $w_v^{(v-1)}(x) = -\sum_{\varepsilon=0}^{v-1} F_\varepsilon(x)$, $\deg w_v^{(v-1)}(x) = \deg F_{\varepsilon'}(x)$. Furthermore,

$$\begin{aligned}\deg \mathcal{X}_v|_v^{(v-1)} &= \deg(w_v^{(v-1)}(x) \cdot x^{(v-1)(k-1)}) \\ &= \deg F_{\varepsilon'}(x) + (v-1)(k-1) \\ &> v(k-1).\end{aligned}$$

TABLE I
PROGRESSIVE ITERATION COMPLEXITY

v	RS (63, 31)		RS (63, 55)	
	image expansion	image reduction	image expansion	image reduction
1	1.53×10^4	4.34×10^3	1.59×10^4	1.02×10^3
2	4.40×10^4	4.55×10^4	4.39×10^4	9.62×10^3
3	1.35×10^5	2.07×10^5	1.43×10^5	4.31×10^4
4	3.48×10^5	6.58×10^5	4.20×10^5	1.50×10^5

Therefore, $\mathrm{LP}(\mathcal{X}_v|_v) \neq v$ and $\mathcal{X}_v$ is not in weak Popov form. ■

Lemma 11 implies the following operation guidance for both the PASD-MM and the ASD-MM algorithms. For the PASD-MM algorithm, if $\deg F_0(x) > k-1$, $\mathcal{X}_1$ will not be in the weak Popov form, neither will the following progressively expanded matrices $\mathcal{X}_2, \mathcal{X}_3$ and etc. Consequently, image reduction needs to be performed in every progressive iteration. For the ASD-MM algorithm that functions with a decoding parameter l, if $|\Upsilon_\varepsilon| = n$ and $\deg F_\varepsilon(x) < k$ for $\varepsilon = 0, 1, \ldots, l-1$, then matrix $\mathcal{A}_l$ will be in weak Popov form. Hence, the formulated module basis $\mathcal{B}_l$ is the intended Gröbner basis. The following basis reduction can be skipped.

VI. COMPLEXITY ANALYSIS

This section analyzes complexity of the proposed PASD-MM and the CR-PASD-MM algorithms. The complexity refers to the number of finite field multiplications needed to decode a codeword. Note that during the decoding, multiplication dominates the finite field arithmetic operations.

A. Complexity of Image Expansion and Reduction

Complexity of image expansion and reduction are characterized by the following two Lemmas, respectively.

Lemma 12: At progressive iteration v, complexity of image expansion is $O(n^2v^3)$.

Proof: The image expansion complexity is measured by the number of multiplications in computing generators (13) and (14).

We fisrt determine complexity of computing (13), which needs to characterize $\max\{\deg_x \mathcal{P}'_{v-1,t}(x,y)\}$. Note that after image reduction, $\max\{\deg_x \mathcal{P}'_{v-1,t}(x,y)\} \leq \max\{\deg_x \mathcal{P}_{v-1,t}(x,y)\}$. In order to simplify the analysis, assume that the image reduction is not performed after each expansion, so that we have $\mathcal{P}_{v-1,t}(x,y) = \prod_{\epsilon=t}^{v-2} R_\epsilon(x) \cdot W_t(x,y)$. Since $\deg R_\epsilon(x) \leq n$ and $\deg_x W_t(x,y) \leq (n-1)t$, we have $\deg_x \mathcal{P}_{v-1,t}(x,y) \leq n(v-t-1) + (n-1)t = n(v-1) - t$. Hence, $\max\{\deg_x \mathcal{P}'_{v-1,t}(x,y)\} \leq n(v-1)$. Therefore, constructing the first v image generators requires at most $\sum_{t=0}^{v-1} n(v-1) \cdot n \cdot v = n^2v^2(v-1)$ multiplications. Note that the naive polynomial multiplication is used.

In computing (14), n^2 multiplications are needed to construct $F_{v-1}(x)$. Since $W_v(x,y) = (y - F_{v-1}(x))W_{v-1}(x,y)$ and $\deg_x W_{v-1}(x,y) \leq (n-1)(v-1)$, complexity of computing $W_v(x,y)$ is n^2v^2. Therefore, complexity of image expansion at progressive iteration v is $\mathcal{C}_{\exp}(v) = n^2v^3 + n^2 + n^2v^2$. Asymptotically, it is $O(n^2v^3)$. ■

Lemma 13: At progressive iteration v, complexity of image reduction is $O(n(n-k)v^4)$.

Proof: The image reduction complexity is determined by $\max\{\deg \mathcal{X}_v|_t^{(\tau)}\}$ and the number of row operations that is required to reduce $\mathcal{X}_v$ into $\mathcal{X}'_v$.

We first characterize $\max\{\deg \mathcal{X}_v|_t^{(\tau)}\}$. After image reduction, we have $\max\{\deg \mathcal{X}'_v|_t^{(\tau)}\} \leq \max\{\deg \mathcal{X}_v|_t^{(\tau)}\}$. Similar to the proof of Lemma 12, assume that image reduction is not performed. Hence, entry of matrix $\mathcal{X}_v$ can be represented as $\mathcal{X}_v|_t^{(\tau)} = \sum_{\epsilon=t}^{v-1} R_\epsilon(x) \cdot w_t^{(\tau)}(x) \cdot x^{(k-1)\tau}$. Since $\deg R_\epsilon(x) \leq n$ and $\deg w_t^{(\tau)}(x) \leq (n-1)(t-\tau)$, we have

$$\begin{aligned}\deg \mathcal{X}_v|_t^{(\tau)} &\leq n(v-t) + (n-1)(t-\tau) + (k-1)\tau \\ &= nv - t - (n-k)\tau.\end{aligned}$$

Therefore, $\max\{\deg \mathcal{X}_v|_t^{(\tau)}\} \leq nv$.

Given a matrix $\mathcal{X}_v$ over $\mathbb{F}_q[x]$, there are less than $v(\mathrm{mdeg}\, \mathcal{X}_v - \deg\det \mathcal{X}_v + v)$ row operations to reduce it into weak Popov form $\mathcal{X}'_v$ [38]. We determine $\mathrm{mdeg}\, \mathcal{X}_v - \deg\det \mathcal{X}_v$ as follows. Let $\tau' = \mathrm{LP}(\mathcal{X}_v|_v)$, we have $\mathrm{rdeg}\, \mathcal{X}_v|_v = \deg w_v^{(\tau')}(x) \cdot x^{(k-1)\tau'}$. Based on (13) – (15),

$$\mathrm{mdeg}\, \mathcal{X}_v = \mathrm{mdeg}(R_{v-1}(x) \cdot \mathcal{X}'_{v-1}) + \deg w_v^{(\tau')}(x) \cdot x^{(k-1)\tau'}.$$

Furthermore,

$$\deg\det \mathcal{X}_v = \deg\det(R_{v-1}(x) \cdot \mathcal{X}'_{v-1}) + \deg x^{(k-1)v}.$$

Since $\mathcal{X}'_{v-1}$ is in weak Popov form, based on Lemma 6, $\mathrm{mdeg}\, \mathcal{X}'_{v-1} = \deg\det \mathcal{X}'_{v-1}$. Hence,

$$\begin{aligned}&\mathrm{mdeg}\, \mathcal{X}_v - \deg\det \mathcal{X}_v \\ &= \deg w_v^{(\tau')}(x) \cdot x^{(k-1)\tau'} - \deg x^{(k-1)v} \\ &\leq (n-1)(v-\tau') - (k-1)(v-\tau').\end{aligned}$$

Therefore, when $\tau' = 0$, $\max\{\mathrm{mdeg}\, \mathcal{X}_v - \deg\det \mathcal{X}_v\} = (n-k)v$. As a result, there are at most $(n-k+1)v^2$ row operations in the image reduction. Since $\max\{\deg \mathcal{X}_v|_t^{(\tau)}\} \leq nv$ and there are $v+1$ entries in each row, complexity of image reduction at progressive iteration v is $\mathcal{C}_{\mathrm{red}}(v) = (n-k+1)nv^3(v+1)$. Asymptotically, it is $O((n-k)nv^4)$. ■

Table I shows the numerical results of each progressive iteration complexity in decoding the (63, 31) and the (63, 55) RS codes. They verify the above analysis. It also shows that different rate codes have a similar complexity for image expansion, while the high rate code exhibits a lower image reduction complexity.

B. Average Complexity of the PASD-MM Algorithm

The proposals can adjust the decoding computation to the received information, recovering the message with the smallest decoding parameter. If the channel condition improves, the received information would be more reliable. As a result, the decoding can terminate with a smaller decoding parameter, resulting in a lower computational cost. Therefore, the progressive decoding complexity is channel dependent. To show this channel dependent feature, we measure the average decoding complexity over multiple decoding events at a

TABLE II
ASYMPTOTIC COMPLEXITY COMPARISON

Algorithm	Complexity
ASD (Koetter) [9]	$O(l^5n^2)$
PASD (Koetter) [34]	$O(l^5n^2)$
MT-GS (Alekhnovich) [27]	$O(l^4mn\log^2 n\log\log n)$
ASD-MM (MS) [25]	$O(l^5n(n-k))$
Chowdhury *et al.* [33]	$O(l^2m^2n\log^3\log\log n)$
PASD-MM (MS)	$O(l^5n(n-k))$
PASD-MM (Alekhnovich)	$O(l^5n\log^2 n\log\log n)$

* $(\cdot)$ is the adopted interpolation or basis reduction technique.

certain SNR. Let $\mathsf{P}(v)$ denote the probability of the progressive decoding algorithms (the PASD-MM and the MT-GS as a benchmark) produce the intended message $f(x)$ at iteration v. Moreover, Lemmas 12 and 13 reveal that the progressive iteration v requires at most $\mathcal{C}(v) = \mathcal{C}_{\text{exp}}(v) + \mathcal{C}_{\text{red}}(v)$ multiplications. Therefore, the average complexity $\mathcal{C}_{\text{avg}}$ of the PASD-MM algorithm can be written as

$$\mathcal{C}_{\text{avg}} = \sum_{v=1}^{l}\mathsf{P}(v)(\sum_{v'=1}^{v}\mathcal{C}(v')) + (1-\sum_{v=1}^{l}\mathsf{P}(v))(\sum_{v'=1}^{l}\mathcal{C}(v')),$$

where $1-\sum_{v=1}^{l}\mathsf{P}(v)$ is the probability of decoding failure in which the progressive decoding terminates with $v = l$. Since complexity of root-finding step is $O(n^2v^2)$, it is marginal in comparison to the progressive interpolation complexity. There also exists several asymptotically faster variants with a complexity of $O(v^2n\log^2 n\log\log n)$ [22] [39]. From the perspective of asymtotic complexity, image reduction dominates the decoding complexity. Therefore, when all decoding events terminate with $v = l$, the PASD-MM algorithm exhibits a worst case complexity, i.e., $O(l^5n(n-k))$ with the MS algorithm. The Alekhnovich algorithm [22] reduces the asymptotic complexity to $O(l^5n\log^2 n\log\log n)$. However, it should be pointed out that this low-complexity image reduction only becomes effective when the codeword length is very large, e.g., beyond 4000 [27]. For practical codes, the MS algorithm remains efficient. Table II compares complexity of the proposed algorithm with some known algebraic decoding algorithms for RS codes. The interpolation or basis reduction techniques are also given. Note that for the progressive decoding algorithms, we consider their worst case complexity. i.e., when $v = l$. It can be seen that the worst case complexity of the progressive algorithms remains the same as their non-progressive variant with the same interpolation or basis reduction technique. However, complexity advantage of the progressive algorithms will become obvious when the channel condition improves, in which the decoding can terminate earlier.

Let d_e denote the number of hard-decision errors in a received word $\underline{\omega}$. At an SNR, its average over all decoding events is denoted as $\tilde{d}_e$ which is in decimal. Table III shows the statistics of $\mathsf{P}(v)$ and $\tilde{d}_e$ in decoding the (63, 31) RS code. $\mathsf{P}(v)$ of the MT-GS algorithm is also shown for comparison. The results were obtained in the additive white Gaussian noise (AWGN) channel with BPSK modulation by running 10 000 decoding events at each SNR. Note that the MT-GS algorithm can correct 16 and 17 errors with a $\deg_y Q$ of 1 and 4, respectively. Table III shows as the SNR increases, there are less hard-decision errors. For both of the progressive decoding algorithms, more decoding events can be terminated earlier, lowering the average complexity. At 7.0 dB, all decoding events were terminated after the first iteration for both of the algorithms. Complexity of the two algorithms converges to the minimum level that is characterized by performing the ASD-MM or the GS algorithm with $l = 1$.

Table IV further shows the average complexity in decoding the (63, 31) RS code. For this code, complexity of the BM and the GMD algorithms are 1.41×10^4 and 8.25×10^5, respectively. In Table IV, the ASD and the PASD algorithms employ Koetter's interpolation [7], in which the PASD algorithm exhibits a memory cost (measured as the number of polynomial coefficients that need to be memorized) of $O(n^2l^4)$ [35]. All algorithms decode with $l = 4$. As SNR increases, the progressive decoding algorithms can decode the message at an earlier iteration, resulting in a lower computational cost. When SNR is sufficiently high, e.g., 7.0 dB, the PASD-MM algorithm yields a complexity reduction of two orders of magnitude over the ASD-MM algorithm. This is similar in comparing the PASD and the ASD algorithms. Moreover, low-complexity feature of the MM interpolation can also be validated by comparing the PASD-MM and the PASD algorithms, as well as the ASD-MM and the ASD algorithms. Table IV also shows that when in the medium SNR regions, the PASD-MM algorithm is less complex than its hard-decision counterpart, the MT-GS algorithm. Even though the progressive algorithms may perform multiple root-finding processes if they terminate at the decoding iteration greater than one, our results show this extra computation can be offset by the progressive interpolation. For this code, our simulation shows about 10% of the decoding computation is spent on root-finding. It is also interesting to note that when both the PASD-MM and the ASD-MM algorithms function with the maximum decoding parameter l, the former is still less complex. As discussed in Section IV.A, the PASD-MM algorithm can be interpreted as rescheduling the row operations of the ASD-MM algorithm. For the PASD-MM algorithm, its basis entries have lower degree than those of the corresponding basis that is handled by the ASD-MM algorithm. This leads to a lower computational cost. Finally, Table IV also shows complexity of the hybrid decoding [40], where Hybrid-1 incorporates the BM and the ASD-MM algorithms and Hybrid-2 incorporates the BM and the PASD-MM algorithms. The ASD-MM (or the PASD-MM) algorithm will only be deployed when the BM algorithm fails. The hybrid decoding systems have further lower complexity which is also channel dependent. Note that at 7.0 dB, the hybrid decoding complexity is defined by that of the BM algorithm (BM decoding donimates). The PASD-MM algorithm exhibits the same complexity magnitude as the BM algorithm.

C. Average Complexity of the CR-PASD-MM Algorithm

Table V further compares complexity of the ASD-MM, the PASD, the PASD-MM and the CR-PASD-MM algorithms in decoding the popular (255, 239) RS code. For this code, complexity of the BM and the GMD algorithms are 4.44×10^4 and 4.65×10^5, respectively. Compared to the ASD-MM

TABLE III

THE STATISTICS OF $\mathsf{P}(v)$ AND $\tilde{d}_e$ IN DECODING THE $(63, 31)$ RS CODE $(l = 4)$

	SNR (dB) P_v (%)	3.0	3.5	4.0	4.5	5.0	5.5	6.0	6.5	7.0
PASD-MM	P_1	1.56	8.97	26.91	58.03	85.29	97.12	99.70	99.98	100
	P_2	2.48	8.75	17.79	17.09	8.51	1.99	0.20	0.02	0
	P_3	2.96	8.56	12.51	9.25	3.10	0.47	0.07	0	0
	P_4	3.77	10.30	12.83	7.86	2.01	0.28	0.02	0	0
MT-GS [27]	P_1	1.42	8.21	26.38	57.19	84.54	97.12	99.70	99.98	100
	P_4	1.36	4.54	9.80	10.96	6.48	1.59	0.20	0.02	0
$\tilde{d}_e$		24.89	21.89	18.93	16.04	13.33	10.77	8.50	6.53	4.83

TABLE IV

AVERAGE COMPLEXITY IN DECODING THE $(63, 31)$ RS CODE $(l = 4)$

SNR (dB)	3.0	3.5	4.0	4.5	5.0	5.5	6.0	6.5	7.0
ASD [9]	9.41×10^6	1.11×10^7	1.27×10^7	1.42×10^7	1.53×10^7	1.59×10^7	1.59×10^7	1.54×10^7	1.48×10^7
PASD [34]	2.37×10^6	2.24×10^6	1.47×10^6	6.44×10^5	2.47×10^5	1.26×10^5	1.06×10^5	1.02×10^5	1.02×10^5
MT-GS* [27]	1.16×10^6	1.09×10^6	8.86×10^5	5.42×10^5	2.27×10^5	7.06×10^4	3.60×10^4	3.20×10^4	3.11×10^4
ASD-MM [25]	1.91×10^6	2.03×10^6	2.12×10^6	2.15×10^6	2.08×10^6	1.98×10^6	1.81×10^6	1.65×10^6	1.51×10^6
PASD-MM	1.49×10^6	1.31×10^6	8.47×10^5	4.14×10^5	1.37×10^5	5.01×10^4	3.33×10^4	3.18×10^4	3.11×10^4
Hybrid-1** [40]	1.90×10^6	1.90×10^6	1.61×10^6	1.01×10^6	4.06×10^5	9.63×10^4	2.55×10^4	1.63×10^4	1.41×10^4
Hybrid-2**	1.48×10^6	1.30×10^6	8.40×10^5	3.87×10^5	1.16×10^5	3.48×10^4	1.96×10^4	1.61×10^4	1.41×10^4

* The decoding trial is $(1, 1, 16) \rightarrow (2, 2, 16) \rightarrow (3, 3, 16) \rightarrow (3, 4, 17)$ where the parameters are the multiplicity, the output list size and the decoding radius, respectively.

** The Hybrid systems incorporate the BM algorithm with either the ASD-MM algorithm (denoted as Hybrid-1) or the PASD-MM algorithm (denoted as Hybrid-2).

TABLE V

AVERAGE COMPLEXITY IN DECODING THE $(255, 239)$ RS CODE

	SNR (dB)	5	6	7	8	9	10	11
$l = 4$	ASD-MM [25]	2.92×10^7	2.82×10^7	2.57×10^7	2.31×10^7	2.26×10^7	2.23×10^7	2.20×10^7
	PASD [34]	2.39×10^8	3.48×10^7	1.10×10^7	1.08×10^7	1.08×10^7	1.08×10^7	1.08×10^7
	PASD-MM	2.05×10^7	3.23×10^6	6.94×10^5	6.87×10^5	6.85×10^5	6.85×10^5	6.85×10^5
	CR-PASD-MM	2.05×10^7	3.23×10^6	6.37×10^5	3.77×10^5	2.08×10^5	1.42×10^5	1.42×10^5
$l = 8$	ASD-MM [25]	3.60×10^8	3.42×10^8	3.35×10^8	3.28×10^8	3.23×10^8	3.17×10^8	3.12×10^8
	PASD [34]	2.26×10^9	1.21×10^8	1.15×10^7	1.08×10^7	1.08×10^7	1.08×10^7	1.08×10^7
	PASD-MM	3.28×10^8	3.02×10^7	6.97×10^5	6.87×10^5	6.85×10^5	6.85×10^5	6.85×10^5
	CR-PASD-MM	3.28×10^8	3.01×10^7	6.41×10^5	3.79×10^5	2.08×10^5	1.42×10^5	1.42×10^5

algorithm, the PASD-MM algorithm exhibits a complexity reduction of at least two orders of magnitude at high SNR. For this code, about 15% of the decoding computation is spent on root-finding at each progressive iteration. Based on Section V, we know during the progressive iteration v, if $\deg F_{v-1}(x) < k$ and $\underline{y}^{(v-1)}$ satisfies the ML criterion, the progressive decoding will terminate without performing the image expansion and reduction. Table V shows the CR-PASD-MM algorithm yields a further complexity reduction over the PASD-MM algorithm. For the (255, 239) RS code, the CR-PASD-MM algorithm starts to show its complexity reduction effect when the SNR is greater than 7 dB. Our decoding statistics shows when SNR $=$ 7 dB, 10.05% of the decoding events are terminated by the above criterion. When SNR $=$ 9 dB, 88.52% of the decoding events are terminated earlier in the same manner. These resutls show the effectiveness of degree assessment at high SNR.

VII. DECODING PERFORMANCE

This section shows decoding performance of the proposals. The frame error rate (FER) is obtained over the AWGN channel.

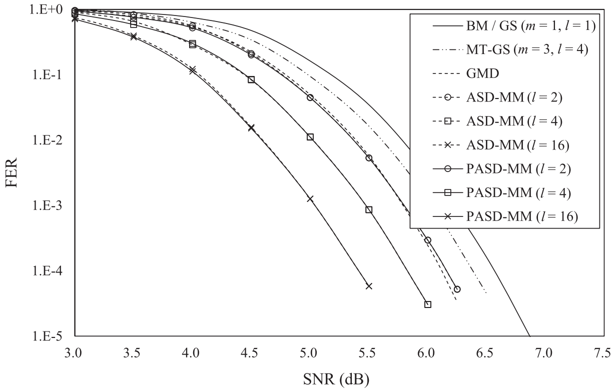


Fig. 1. Performance of the (63, 31) RS code with BPSK modulation.

Fig. 1 shows performance of the (63, 31) RS code using BPSK modulation. It can be seen that the ASD-MM and the PASD-MM algorithms perform the same. They both outperform the BM, the GMD and the MT-GS algorithms. Note that the MT-GS algorithm can correct 16 and 17 errors with $(m = 1, l = 1)$ and $(m = 3, l = 4)$, respectively. Performance of the ASD-MM and the PASD-MM algorithms enhance by

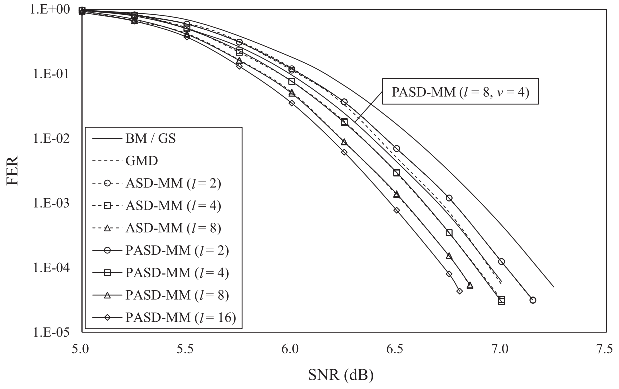


Fig. 2. Performance of the (255, 239) RS code with BPSK modulation.

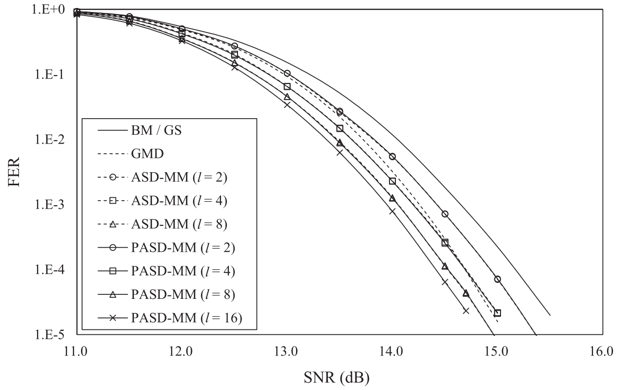


Fig. 3. Performance of the (63, 55) RS code with 64QAM modulation.

increasing l. Revisiting the complexity results of Table IV, we know that over the whole spectrum of SNR, the PASD-MM algorithm is less complex than the ASD-MM algorithm.

Fig. 2 shows performance of the popular (255, 239) RS code again using BPSK. Again, it shows with the same decoding parameter l, the ASD-MM and the PASD-MM algorithms perform the same. Their performances improve with l and outperform the GMD algorithm when $l \geq 4$. Fig. 2 further shows performance of the PASD-MM algorithm with $l = 8$ but terminated at $v = 4$. It means that matrix $\mathbf{M}$ is generated by $l = 8$, but the PASD-MM algorithm only utilizes matrix $\tilde{\mathbf{M}}_1$, $\tilde{\mathbf{M}}_2$, $\tilde{\mathbf{M}}_3$ and $\tilde{\mathbf{M}}_4$ for the progressive decoding. The curve is marked by PASD-MM $(l = 8, v = 4)$. It can be seen that it performs worse than the PASD-MM $(l = 4)$, implying the intermediate $\tilde{\mathbf{M}}_4$ does not approximate $\mathbf{\Pi}$ as well as $\mathbf{M}$ that corresponds to $l = 4$. A better iterative strategy of constructing the intermediate $\tilde{\mathbf{M}}_v$ would help find the intended message earlier. This will be our future work. It should be pointed out that the PASD-MM $(l = 8)$ still achieves the same performance as the ASD-MM $(l = 8)$. Note that the CR-PASD-MM algorithm also maintains the ASD performance. Table V shows the CR-PASD-MM algorithm can further reduce the complexity over the PASD-MM algorithm. It is more effective when the SNR is large. In this scenario, the degree assessment of Lagrange interpolation polynomials is more effective for complexity reduction.

Finally, Fig. 3 shows performance of the (63, 55) RS code using 64QAM. It again shows performance of the ASD-MM and the PASD-MM algorithms improve as l increases and approach that of the GMD algorithm when $l = 4$. Compared with the BM algorithm, the algebraic decoding with $l = 16$ yields a performance gain of 0.8 dB at the FER of 10^{-4}.

VIII. Conclusion

This paper has introduced the PASD-MM algorithm for RS codes, which is a progressive embodiment of the ASD-MM algorithm and a soft-decision extension of the MT-GS algorithm. It produces the interpolation polynomial with a progressively enlarged y-degree, adjusting the error-correction capability and decoding computation to the received information. It has been shown using the MM technique, the progressive interpolation can be realized through the image of the progressively enlarged submodule basis. Our validity analysis has demonstrated that finding message from the submodule basis is equivalent to finding it from its image. Furthermore, a complexity reducing variant of the PASD-MM algorithm has been proposed based on assessing the degree of Lagrange interpolation polynomials. Complexity analysis of the proposed algorithms has also been performed. Our simulation results have shown that significant complexity reduction can be achieved and the proposed algorithms maintain the ASD error-correction capability. This work is an advancement over the original PASD approach since the progressive MM interpolation is realized without any additional memory cost. This feature can facilitate the application of the progressive RS decoding.

Appendix

A. The ML Criterion

With the reliability matrix $\mathbf{\Pi}$, we can identify the largest and the second largest entries of column j as $\pi_j^{\mathrm{I}} = \max\{\pi_{ij}, \forall i\}$ and $\pi_j^{\mathrm{II}} = \max\{\pi_{ij}, \forall i \text{ and } \pi_{ij} \neq \pi_j^{\mathrm{I}}\}$, respectively. Given a hard-decision received word $\underline{\omega} = (\omega_0, \omega_1, \ldots, \omega_{n-1})$ and an estimated codeword $\underline{\hat{c}} = (\hat{c}_0, \hat{c}_1, \ldots, \hat{c}_{n-1})$, we define

$$\Omega_1(\underline{\omega}, \underline{\hat{c}}) = \prod_{j:\hat{c}_j \neq \omega_j} \frac{\pi_j^{\mathrm{I}}}{\pi_{\hat{i}_j j}},$$

where $\hat{i}_j = \text{index}\{\sigma_i \mid \sigma_i = \hat{c}_j\}$. For the codeword symbol positions where $\hat{c}_j = \omega_j$, sort elements of the set $\{\frac{\pi_j^{\mathrm{I}}}{\pi_j^{\mathrm{II}}}, \forall \hat{c}_j = \omega_j\}$ in an ascending order such as

$$\frac{\pi_{j_0}^{\mathrm{I}}}{\pi_{j_0}^{\mathrm{II}}} \leq \frac{\pi_{j_1}^{\mathrm{I}}}{\pi_{j_1}^{\mathrm{II}}} \leq \cdots .$$

We can further define

$$\Omega_2(\underline{\omega}, \underline{\hat{c}}) = \prod_{\zeta=0}^{d_{\min}-d-1} \frac{\pi_{j_\zeta}^{\mathrm{I}}}{\pi_{j_\zeta}^{\mathrm{II}}},$$

where $d_{\min} = n - k + 1$ is the minimum Hamming distance of the code and d is the Hamming distance between $\underline{\omega}$ and $\underline{\hat{c}}$. If

$$\Omega_1(\underline{\omega}, \underline{\hat{c}}) \leq \Omega_2(\underline{\omega}, \underline{\hat{c}}),$$

then $\underline{\hat{c}}$ is the ML codeword [37].

References

[1] E. Berlekamp, *Algebraic Coding Theory*. New York, NY, USA: McGraw-Hill, 1968.

[2] J. Massey, "Shift-register synthesis and BCH decoding," *IEEE Trans. Inf. Theory*, vol. IT-15, no. 1, pp. 122–127, Jan. 1969.

[3] L. Welch and E. Berlekamp, "Error correction for algebraic block codes," U.S. Patent 4 633 470 A, Dec. 12, 1986.

[4] R. Kotter, "Fast generalized minimum-distance decoding of algebraic-geometry and Reed–Solomon codes," *IEEE Trans. Inf. Theory*, vol. 42, no. 3, pp. 721–737, May 1996.

[5] M. Sudan, "Decoding of Reed–Solomon codes beyond the error-correction bound," *J. Complex.*, vol. 13, no. 1, pp. 180–193, Mar. 1997.

[6] V. Guruswami and M. Sudan, "Improved decoding of Reed–Solomon and algebraic-geometric codes," *IEEE Trans. Inf. Theory*, vol. 45, no. 1, pp. 1757–1767, Mar. 1999.

[7] R. Koetter, "On algebraic decoding of algebraic-geometric and cyclic codes," Ph.D. dissertation, Linköping Studies Sci. Technol., Linköping Univ., Linköping, Sweden, 1996.

[8] R. Roth and G. Ruckenstein, "Efficient decoding of Reed–Solomon codes beyond half the minimum distance," *IEEE Trans. Inf. Theory*, vol. 46, no. 1, pp. 246–257, Jan. 2000.

[9] R. Koetter and A. Vardy, "Algebraic soft-decision decoding of Reed–Solomon codes," *IEEE Trans. Inf. Theory*, vol. 49, no. 11, pp. 2809–2825, Nov. 2003.

[10] M. El-Khamy, R. J. McEliece, and J. Harel, "Performance enhancements for algebraic soft decision decoding of Reed–Solomon codes," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Chicago, lL, USA, Jun. 2004, p. 421.

[11] H. Das and A. Vardy, "Multiplicity assignments for algebraic soft-decoding of Reed–Solomon codes using the method of types," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Seoul, South Korea, Jun. 2009, pp. 1248–1252.

[12] J. Bellorado and A. Kavcic, "Low-complexity soft-decoding algorithms for Reed–Solomon codes—Part I: An algebraic soft-in hard-out chase decoder," *IEEE Trans. Inf. Theory*, vol. 56, no. 3, pp. 945–959, Mar. 2010.

[13] R. Koetter and A. Vardy, "A complexity reducing transformation in algebraic list decoding of Reed–Solomon codes," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Paris, France, Mar. 2003, pp. 10–13.

[14] R. Koetter, J. Ma, and A. Vardy, "The re-encoding transformation in algebraic list-decoding of Reed–Solomon codes," *IEEE Trans. Inf. Theory*, vol. 57, no. 2, pp. 633–647, Feb. 2011.

[15] J. Ma, P. Trifonov, and A. Vardy, "Divide-and-conquer interpolation for list decoding of Reed–Solomon codes," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Chicago, IL, USA, Jun. 2004, p. 386.

[16] J. S. R. Nielsen, "Fast Kötter–Nielsen–Høholdt interpolation in the Guruswami-Sudan algorithm," in *Proc. 14th Int. Workshop Alg. Combinat. Coding Theory (ACCT)*, Svetlogorsk, Russia, Sep. 2014, pp. 1–6.

[17] Y. Cassuto, J. Bruck, and R. J. McEliece, "On the average complexity of Reed–Solomon list decoders," *IEEE Trans. Inf. Theory*, vol. 59, no. 4, pp. 2336–2351, Apr. 2013.

[18] K. Lee and M. O'Sullivan, "List decoding of Reed–Solomon codes from a Gröbner basis perspective," *J. Symbolic Comput.*, vol. 43, no. 9, pp. 645–658, Feb. 2008.

[19] D. A. Cox, J. Little, and D. O'Shea, *Using Algebraic Geometry*, 2nd ed. New York, NY, USA: Springer, 2005.

[20] T. Mulders and A. Storjohann, "On lattice reduction for polynomial matrices," *J Symbolic Comput.*, vol. 35, no. 4, pp. 377–401, Apr. 2003.

[21] P. Giorgi, C.-P. Jeannerod, and G. Villard, "On the complexity of polynomial matrix computations," in *Proc. ACM Int. Symp. Symbolic Algebric Comput. (ISSAC)*, Aug. 2003, pp. 135–142.

[22] M. Alekhnovich, "Linear diophantine equations over polynomials and soft decoding of Reed–Solomon codes," *IEEE Trans. Inf. Theory*, vol. 51, no. 7, pp. 2257–2265, Jul. 2005.

[23] S. Gupta, S. Sarkar, A. Storjohann, and J. Valeriote, "Triangular x-basis decompositions and derandomization of linear algebra algorithms over K[x]," *J. Symbolic Comput.*, vol. 47, no. 4, pp. 422–453, Apr. 2012.

[24] C.-P. Jeannerod, V. Neiger, É. Schost, and G. Villard, "Computing minimal interpolation bases," *J. Symbolic Comput.*, vol. 83, pp. 272–314, Nov./Dec. 2017.

[25] K. Lee and M. E. O'Sullivan, "An interpolation algorithm using Gröbner bases for soft-decision decoding of Reed–Solomon codes," in *Proc. IEEE Int. Symp. Inf. Theory*, Seattle, WA, USA, Jul. 2006, pp. 2032–2036.

[26] J. Ma and A. Vardy, "A complexity reducing transformation for the Lee-O'Sullivan interpolation algorithm," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Nice, France, Jun. 2007, pp. 1986–1990.

[27] J. S. R. Nielsen and A. Zeh, "Multi-trial Guruswami–Sudan decoding for generalised Reed–Solomon codes," *Des. Codes Cryptogr.*, vol. 73, no. 2, pp. 507–527, Nov. 2014.

[28] L. Chen and M. Bossert, "Algebraic chase decoding of Reed–Solomon codes using module minimisation," in *Proc. Int. Symp. Inf. Theory App. (ISITA)*, Monterey, CA, USA, Oct. 2016, pp. 305–309.

[29] J. Rosenkilde, "Power decoding Reed–Solomon codes up to the Johnson radius," *Adv. Math. Commun.*, vol. 12, no. 1, pp. 81–106, Feb. 2018.

[30] P. V. Trifonov, "Efficient interpolation in the Guruswami–Sudan algorithm," *IEEE Trans. Inf. Theory*, vol. 56, no. 9, pp. 4341–4349, Aug. 2010.

[31] V. Miloslavskaya and P. Trifonov, "Fast interpolation in algebraic soft decision decoding of Reed–Solomon codes," in *Proc. IEEE Int. Conf. Comp. Tech. Elect. Electron. Eng. (SIBIRCON)*, Listvyanka, Russia, Jul. 2010, pp. 65–69.

[32] V. Miloslavskaya and P. Trifonov, "Hybrid interpolation algorithm for algebraic soft decision decoding of Reed–Solomon codes," in *Proc. 8th Int. Symp. Wireless Commun. Syst.*, Aachen, Germany, Nov. 2011, pp. 131–135.

[33] M. F. I. Chowdhury, C.-P. Jeannerod, V. Neiger, É. Schost, and G. Villard, "Faster algorithms for multivariate interpolation with multiplicities and simultaneous polynomial approximations," *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2370–2387, May 2015.

[34] 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.

[35] Y. Lyu and L. Chen, "Algebraic soft decoding of Reed–Solomon codes with improved progressive interpolation," *Phys. Commun.*, vol. 20, pp. 48–60, Sep. 2016.

[36] H. Hasse, "Theorie der höheren differentiale in einem algebraischen funktionenkörper mit vollkommenem konstantenkörper bei beliebiger charakteristik," *J. Die Reine Angew. Math.*, vol. 175, pp. 50–54, 1936.

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

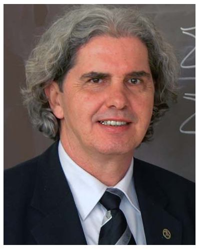
[38] J. Nielsen, "List decoding algebraic codes," Ph.D. dissertation, Dept. Appl. Math. Comput. Sci., Tech. Univ. Denmark, Lyngby, Denmark, 2013.

[39] V. Neiger, J. Rosenkilde, and É. Schost, "Fast computation of the roots of polynomials over the ring of power series," in *Proc. ACM Int. Symp. Symbolic Algebr. Comput.*, Kaiserslautern, Germany, Jul. 2017, pp. 349–356.

[40] H. Xia, H. Song, and J. R. Cruz, "Retry mode soft Reed–Solomon decoding," *IEEE Trans. Magn.*, vol. 38, no. 5, pp. 2325–2327, Sep. 2002.

Jiongyue Xing (S'17) received the B.Sc. degree in communication engineering from Sun Yat-sen University, Guangzhou, China, in 2015, where he is currently pursuing the Ph.D. degree in information and communication engineering. He is also a Visiting Ph.D. Student at the Institute of Communication Engineering, Ulm University, Germany. His research interests include channel coding and data communications.

Li Chen (S'07–M'08–SM'14) received the B.Sc. degree in applied physics from Jinan University, China, in 2003, and the M.Sc. degree in communications and signal processing and the Ph.D. degree in communications engineering from Newcastle University, U.K., in 2004 and 2008, respectively. From 2007 to 2010, he was a Research Associate with Newcastle University. In 2010, he was a Lecturer with the School of Information Science and Technology, Sun Yat-sen University, China, where he became an Associate Professor, in 2011. He has been a Professor with the School of Electronics and Communication Engineering, Sun Yat-sen University, since 2016. He is also the Deputy Dean of the School. From 2011 to 2012, he was an occasional Visiting Scholar with the Institute of Network Coding, The Chinese University of Hong Kong. In 2015, he was a Visitor at the Institute of Communication Engineering, Ulm University, Germany. From 2015 to 2016, he was a Visiting Associate Professor with the Department of Electrical Engineering, University of Notre Dame, Notre Dame, IN, USA. His primary research interests include information theory, channel coding, and data communications. He is a member of the Conference Committee of IEEE Information Theory Society as well as the Chinese Information Theory Society. He is also a Senior Member of the Chinese Institute of Electronics. He has been a Technical Program Committee (TPC) Member of various international conferences. As the General Co-Chair, he has hosted the 2018 IEEE Information Theory Workshop (ITW) at Guangzhou. He was a recipient of the British Overseas Research Scholarship and the Chinese Information Theory Young Researcher in 2014. He has been a Principle Investigator for three National Natural Science Foundation of China projects and a Co-Investigator of a National Basic Research Program (973 program) Project. He is an Associate Editor of IEEE TRANSACTIONS ON COMMUNICATIONS.



Martin Bossert (M'94–SM'03–F'12) received the Dipl.Ing. degree in electrical engineering from the Technical University of Karlsruhe, Germany, in 1981, and the Ph.D. degree from the Technical University of Darmstadt, Germany, in 1987. After a one-year DFG Scholarship at Linköping University, Sweden, he joined AEG Mobile Communication, where he was involved in the specification and development of the GSM system. Since 1993, he has been a Professor with Ulm University, Germany, where he is currently the Director of the Institute of Communications Engineering. He is the author of several textbooks and the coauthor of more than 200 papers. His research interest includes reliable and secure data transmission. His main focus is on the decoding of algebraic codes with reliability information and coded modulation. He was a member of the IEEE Information Theory Society Board of Governors from 2010 to 2012, and was appointed as a member of the German National Academy of Sciences, Leopoldina, in 2013. Among other awards and honors, he received the Vodafone Innovationspreis 2007.