Low-Complexity Chase Decoding of Hermitian Codes With Improved Interpolation and Root-Finding

Jiwei Liang, Jianguo Zhao, and Li Chen, *Senior Member, IEEE*

***Abstract*— This paper proposes the low-complexity Chase (LCC) decoding for Hermitian codes, which is facilitated by both the improved interpolation and root-finding. By identifying η unreliable received symbols, 2^η test-vectors are formulated, each of which is decoded by the interpolation based Guruswami-Sudan (GS) algorithm. To reduce both the interpolation complexity and latency, the re-encoding transform (ReT) is introduced through defining the Lagrange interpolation polynomials over the Hermitian function fields. The interpolation polynomial is further computed through module basis reduction (BR) that yields the Gröbner basis that contains the desired polynomial. The BR interpolation exhibits a greater parallelism than the conventional Kötter's interpolation. Moreover, the 2^η root-finding processes are facilitated by estimating the codewords directly from the interpolation outcomes. It eliminates the re-encoding computation for identifying the most likely candidate from the decoding output list. It is also shown that the average LCC decoding complexity can be further reduced by both assessing the re-encoding outcome and decoding the test-vectors progressively. They can achieve an early decoding termination once a codeword that satisfies the maximum likelihood (ML) criterion is found. Our simulation results demonstrate that the decoding complexity and latency can be significantly reduced over the existing decoding algorithms.**

***Index Terms*— Basis reduction, early termination, fast root-finding, Hermitian codes, re-encoding transform.**

I. Introduction

ALGEBRAIC-GEOMETRIC (AG) codes, which were first introduced by Goppa [1], are a class of linear block codes derived from an algebraic curve. AG codes comprise a large family, including Reed-Solomon (RS) codes, elliptic codes, Hermitian codes, and etc. The widely used RS codes can be viewed as a special subclass of AG codes since they are constructed from a straight line. However, the length of an RS code cannot exceed the size of the finite field in which it is defined, limiting its error-correction capability. Compared with RS codes, general AG codes have greater codeword lengths, leading to a stronger error-correction capability. Therefore, AG codes have the potential to replace RS codes in many practical communication and storage systems.

Similar to RS codes, AG codes can be decoded by the syndrome-based decoding algorithms, in which the error-correction is realized by determining the error locations and error magnitudes, respectively. The Berlekamp-Massey (BM) algorithm [2] and the Sakata algorithm [3] are well known syndrome-based decoding algorithms for RS and Hermitian codes, respectively. They exhibit a complexity of $O(n^2)$ and $O(n^{7/3})$, respectively, where n is the codeword length. But their error-correction capabilities are limited by half of the code's minimum Hamming distance. The Guruswami-Sudan (GS) algorithm can correct errors beyond this half distance bound by formulating the decoding as a curve-fitting problem [4]. In comparison with syndrome-based decoding, the GS algorithm can yield a significantly improved decoding performance by increasing the interpolation multiplicity. The GS algorithm consists of interpolation and root-finding, where the former dominates the decoding complexity. It constructs a minimum polynomial which has a zero of multiplicity m over a set of interpolation points. The root-finding determines the z-root of the minimum polynomial as the estimated message, which is usually realized by the recursive coefficient search (RCS) algorithm [5], [6], [7]. For the interpolation, it can be realized by Kötter's algorithm [8] which constructs the interpolation polynomial by interpolating n interpolation points iteratively. Interpolation can also be realized from the perspective of the Gröbner basis of modules. The module basis comprises a set of polynomials satisfying the interpolation multiplicity and degree constraints. It can be initially constructed and then reduced into the Gröbner basis, from which the desired interpolation polynomial can be retrieved. This is called the basis reduction (BR) interpolation. Lee and O'Sullivan presented efficient BR interpolation algorithms for both RS codes [9] and Hermitian codes [10]. Based on polynomial-ring matrix minimization, Nielsen and Beelen proposed a fast realization of BR interpolation for Hermitian codes [11], yielding a sub-quadratic complexity in n. Beelen et al. further generalized and improved the

Received 11 August 2024; revised 29 October 2024; accepted 22 December 2024. Date of publication 30 December 2024; date of current version 18 August 2025. This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 62071498; and in part by the Natural Science Foundation of Guangdong Province under Grant 2024A1515010213. An earlier version of this paper was presented in part at the 2023 IEEE Globecom Workshops, Kuala Lumpur, Malaysia [DOI: 10.1109/GCWkshps58843.2023.10464872]. The associate editor coordinating the review of this article and approving it for publication was H. Mahdavifar. *(Corresponding author: Li Chen.)*

Jiwei Liang and Jianguo Zhao are with the School of System Science and Engineering, Sun Yat-sen University, Guangzhou 510006, China (e-mail: liangjw59@mail2.sysu.edu.cn; zhaojg5@mail2.sysu.edu.cn).

Li Chen is with the School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou 510006, China, and also with Guangdong Province Key Laboratory of Information Security Technology, Guangzhou 510006, China (e-mail: chenli55@mail.sysu.edu.cn).

Digital Object Identifier 10.1109/TCOMM.2024.3524035

computation method from [11] for general AG codes [12], [13]. On the other aspect, it has been shown that interpolation can be facilitated by the re-encoding transform (ReT) [14]. It has been applied to reduce the interpolation complexity in decoding RS and elliptic codes [14], [15].

The soft-decision development of GS decoding, i.e., algebraic soft decoding (ASD) algorithm, was introduced by Kötter and Vardy for RS codes [16]. The ASD algorithm enhances the decoding performance by converting reliability information into interpolation multiplicity. Chen et al. [17] and Lee et al. [18] later proposed the ASD algorithm for Hermitian codes using Kötter's interpolation and the BR interpolation, respectively. Chase decoding is another important algebraic soft-decision decoding approach. Although the complexity of Chase decoding is exponential in nature, it often exhibits performance and complexity advantages over other algebraic soft-decision decoding methods, especially in decoding high-rate codes. In Chase decoding, 2^{η} test-vectors can be formulated by identifying η unreliable received symbols. The decoding is performed for each test-vector. By exploiting the similarity among all test-vectors, the low-complexity Chase (LCC) decoding was proposed for RS codes [19] using Kötter's interpolation. The common element interpolation can be performed once for all test-vectors, while the uncommon element interpolation can be further performed in a binary tree growing fashion. With the same test-vectors formulation, LCC decoding of Hermitian codes was proposed by Wu et al. [20]. With a similar low-complexity Kötter's interpolation, it however has not been facilitated by the ReT. Wan et al. [21] proposed the LCC decoding facilitated by the ReT for elliptic codes. However, it imposes a restriction on the number of unreliable positions in forming the test-vectors. Moreover, despite the fact that interpolation usually dominates the complexity of GS decoding, in LCC decoding, the root-finding needs to be performed for 2^{η} decoding events. Its complexity becomes significant as η increases. Therefore, it is also important to reduce the complexity of root-finding in LCC decoding. Meanwhile, the RCS algorithm estimates the message symbols in a serial manner, which inevitably results in a large decoding latency. Addressing this, the factorization-free algorithm [22] that directly computes the estimated codewords from the interpolation outcomes has been introduced for LCC decoding of RS codes. Recently, the ReT-assisted LCC decoding of Hermitian codes was proposed by the authors [23] using Kötter's interpolation. It is also facilitated by the fast factorization (FF) which is generalized from the root-finding algorithm in [22]. However, as mentioned above, Kötter's interpolation cannot process all test-vectors in a fully parallel manner. Both the LCC decoding complexity and latency can be reduced by the more advanced interpolation approach.

This paper proposes the LCC decoding for Hermitian codes, which is facilitated by both the ReT-assisted BR interpolation and the fast root-finding (FRF). The major contributions of this work are summarized as follows:

1) By defining Lagrange interpolation polynomials over the Hermitian function fields, the ReT is introduced to reduce both interpolation complexity and latency. It first shifts the interpolation points based on the code's linear property, and then transforms them by extracting the common factor of module basis polynomials. We also present an improved method to choose the re-encoding points. Therefore, the restriction on the number of unreliable positions in [21] has been lifted. This yields a more flexible formulation of the decoding test-vectors.
2) The ReT-assisted BR interpolation is performed to obtain the interpolation polynomial of each test-vector, enhancing the LCC decoding efficiency for Hermitian codes. The common computation first constructs a partial interpolation module basis. The uncommon computation then completes the module basis construction for each test-vector and reduces them into their respective Gröbner bases. The interpolation polynomials w.r.t. each test-vector can be retrieved from the Gröbner bases. It exhibits a greater parallelism than Kötter's interpolation.
3) The FRF is proposed to facilitate the 2^{η} root-finding processes. It computes the estimated codeword symbols in parallel and removes the need of encoding the estimated messages for identifying the most likely codeword candidate from the decoding output list. Unlike the FF in [23], it makes a better usage of the interpolating polynomial to indicate the error locations, further reducing the decoding complexity.
4) Two strategies for terminating the Chase decoding earlier are proposed, eliminating the redundant computations and adapting the decoding complexity to the reliability of received information. We show that the re-encoding can produce a Hermitian codeword that satisfies the maximum likelihood (ML) criterion [24], [25]. In this case, the decoding can be terminated without executing the following interpolation and root-finding. Meanwhile, decoding of the test-vectors can be processed progressively based on their reliabilities [26], [27]. That says a more reliable test-vector will be decoded prior to the less reliable one. This enables an early termination for the decoding.

The rest of this paper is organized as follows. Section II provides background knowledge for the proposed LCC decoding of Hermitian codes. Section III introduces the test-vectors formulation and the ReT. Section IV introduces the ReT-assisted BR interpolation. Section V introduces the FRF. Section VI introduces early termination strategies for the LCC decoding. The decoding complexity and latency are analysed in Section VII. The decoding performance is demonstrated in Section VIII. Finally, Section IX concludes the paper.

II. Background Knowledge

This section presents the background knowledge of the proposed work, including the encoding of Hermitian codes and the GS decoding algorithm.

A. Hermitian Codes

Let $\mathbb{F}_q = \{\sigma_0, \sigma_1, \ldots, \sigma_{q-1}\}$ denote the finite field of size q. Let $\mathbb{F}_q[X, Y]$ denote the bivariate polynomial ring defined

over $\mathbb{F}_q$. An affine Hermitian curve defined over $\mathbb{F}_q$ can be written as [28]

$$H_w = Y^w + Y - X^{w+1}, \tag{1}$$

where $w = \sqrt{q}$ and the curve has a genus $g = \frac{w(w-1)}{2}$. Note that Hermitian codes are constructed over finite fields with the field size being a square. There are w^3 affine points $P_j = (x_j, y_j)$ that satisfy $H_w(x_j, y_j) = 0$, and a point at infinity P_∞. Let $\mathcal{P}$ denote the set of affine points $\mathcal{P} = \{P_j = (x_j, y_j) \mid H_w(x_j, y_j) = 0\}$, and hence $|\mathcal{P}| = w^3$. The coordinate ring of H_w is

$$R = \mathbb{F}_q[X, Y]/ < Y^w + Y - X^{w+1} > . \tag{2}$$

Let x and y denote the residue classes of X and Y, respectively. The pole basis $\mathscr{L}_w$ of a Hermitian curve comprises a set of bivariate monomials $\phi_a(x, y) = x^{i_x} y^{i_y}$ that exhibit an increasing pole order at P_∞ as $\mathscr{L}_w = \{\phi_a | \upsilon_{P_\infty}(\phi_a^{-1}) < \upsilon_{P_\infty}(\phi_{a+1}^{-1})\}$, where $\upsilon_{P_\infty}(\phi_a^{-1}) = \upsilon_{P_\infty}((x^{i_x} y^{i_y})^{-1}) = w \cdot i_x + (w+1) \cdot i_y$, $0 \le i_x \le w$ and $i_y \ge 0$. For each P_j, there also exists a zero basis comprising polynomials with an increasing zero order (or multiplicity) at the point, the zero basis are [29]

$$\psi_{P_j, v}(x, y) = (x - x_j)^\lambda [(y - y_j) - x_j^w (x - x_j)]^\delta, \tag{3}$$

where $(\lambda, \delta) \in \mathbb{N}$ and $v = \lambda + (w+1)\delta$.

Definition 1: For an (n, k) Hermitian code of length n and dimension k, given a message polynomial

$$f(x, y) = f_0\phi_0 + f_1\phi_1 + \cdots + f_{k-1}\phi_{k-1} \in \mathcal{L}(\mu P_\infty), \tag{4}$$

where $f_0, f_1, \ldots, f_{k-1} \in \mathbb{F}_q$ are the message symbols, $\mu = k + g - 1$ and $\mathcal{L}(\mu P_\infty)$ is the Riemann-Roch space defined by μ and P_∞, the codeword is generated by

$$\underline{c} = (c_0, c_1, \ldots, c_{n-1}) = (f(P_0), f(P_1), \ldots, f(P_{n-1})), \tag{5}$$

where $\{P_0, P_1, \ldots, P_{n-1}\} \subseteq \mathcal{P}$ and its index set is denoted by $[0 : n-1] = \{0, 1, \ldots, n-1\}$.

For an (n, k) Hermitian code, the Riemann-Roch theorem [30] provides the relationship between μ and $\upsilon_{P_\infty}(\phi_{k-1}^{-1})$ as

$$\upsilon_{P_\infty}(\phi_{k-1}^{-1}) \le \mu. \tag{6}$$

B. The GS Decoding

Given a received word $\underline{\omega} = (\omega_0, \omega_1, \ldots, \omega_{n-1}) \in \mathbb{F}_q^n$, the set of interpolation points is denoted by $\mathbf{P} = \{(P_0, \omega_0), (P_1, \omega_1), \ldots, (P_{n-1}, \omega_{n-1})\}$. Let $R[z]$ denote the polynomial ring defined over R. For GS decoding of an (n, k) Hermitian code, the following definitions are needed.

Definition 2: Monomials $\phi_a z^b \in R[z]$ are ordered according to their $(1, w_z)$-weighted degrees that are defined as

$$\deg_{1,w_z} \phi_a z^b = \upsilon_{P_\infty}(\phi_a^{-1}) + w_z b, \tag{7}$$

where $w_z = \upsilon_{P_\infty}(\phi_{k-1}^{-1})$. The $(1, w_z)$-reverse lexicographic (revlex) order can be established as follows. Given two monomials $\phi_{a_1} z^{b_1}$ and $\phi_{a_2} z^{b_2}$, $\mathrm{ord}(\phi_{a_1} z^{b_1}) < \mathrm{ord}(\phi_{a_2} z^{b_2})$, if $\deg_{1,w_z} \phi_{a_1} z^{b_1} < \deg_{1,w_z} \phi_{a_2} z^{b_2}$, or $\deg_{1,w_z} \phi_{a_1} z^{b_1} = \deg_{1,w_z} \phi_{a_2} z^{b_2}$ and $b_1 < b_2$.

Definition 3: Given a polynomial $Q(x, y, z) = \sum_{a,b \in \mathbb{N}} Q_{ab} \phi_a(x, y) z^b$, the $(1, w_z)$-weighted degree of Q is $\deg_{1,w_z} Q = \max\{\deg_{1,w_z} \phi_a z^b \mid Q_{ab} \neq 0\}$ and its leading order is $\mathrm{lod}(Q) = \max\{\mathrm{ord}(\phi_a z^b) \mid Q_{ab} \neq 0\}$.

Note that in decoding an (n, k) Hermitian code, polynomials are organized under the $(1, w_z)$-revlex order. Given two polynomials Q_1 and Q_2, we claim $Q_1 < Q_2$, if $\mathrm{lod}(Q_1) < \mathrm{lod}(Q_2)$.

The interpolation constraint for a polynomial Q in $R[z]$ is explained as follows. Given an interpolation point (P_j, ω_j), if a polynomial Q can be written as

$$Q = \sum_{\alpha,\beta \in \mathbb{N}} Q_{\alpha,\beta}^{(P_j,\omega_j)} \psi_{P_j,\alpha}(z - \omega_j)^\beta, \tag{8}$$

where $Q_{\alpha,\beta}^{(P_j,\omega_j)} \in \mathbb{F}_q$ and $Q_{\alpha,\beta}^{(P_j,\omega_j)} = 0$ for $\alpha + \beta < m$, Q interpolates (P_j, ω_j) with a multiplicity of m. In particular, when $m = 1$, Q's interpolation condition at (P_j, ω_j) can be simplified into its evaluation at the point [20].

Theorem 1 [31]: Given the interpolation polynomial $\mathcal{Q}$ which has a zero of multiplicity m over the n interpolation points, if $m(n - |\{j \mid f(P_j) \neq w_j, \forall j \in [0 : n-1]\}|) > \deg_{1,w_z} \mathcal{Q}$, $\mathcal{Q}(x, y, f) = 0$, or $(z - f) | \mathcal{Q}$.

With a received word $\underline{\omega}$, interpolation constructs the minimum polynomial $\mathcal{Q}$ with respect to the $(1, w_z)$-revlex order. This can be realized by Kötter's interpolation, which starts with a polynomial set and updates its polynomials in an iterative manner. Finally, the minimum polynomial in the set will be chosen as the interpolation polynomial $\mathcal{Q}$. The z-roots of $\mathcal{Q}$ will then be determined as the estimated message.

The interpolation polynomial $\mathcal{Q}$ can also be determined by BR interpolation. It first constructs a basis of the module which contains polynomials that satisfy the interpolation constraints. It will then be reduced into a Gröbner basis [10], from which $\mathcal{Q}$ can be retrieved. Let $R[z]_l = \{Q \in R[z] \mid \deg_z Q \le l\}$, where l is the decoding output list size. For a set of interpolation points $\mathbf{S}$, the interpolation ideal $\mathcal{I}_{\mathbf{S},m}$ is defined as a set of all polynomials over $R[z]$ that interpolate $\mathbf{S}$ with a multiplicity of m. The interpolation module $I_{\mathbf{S},m}$ is defined as $I_{\mathbf{S},m} = \mathcal{I}_{\mathbf{S},m} \cap R[z]_l$. In the proposed LCC decoding, $m = l = 1$. For simplicity, we use $I_{\mathbf{S}}$ instead of $I_{\mathbf{S},1}$.

In order to introduce the BR interpolation, the following definitions can be further introduced.

Definition 4: Given an affine point index set $\mathcal{J} \subseteq [0 : n-1]$, $\mathbb{A}(\mathcal{J}) = \{x_j \mid j \in \mathcal{J}\}$. Given $\sigma \in \mathbb{A}(\mathcal{J})$, $\mathbb{B}_\sigma(\mathcal{J}) = \{y_j \mid (\sigma, y_j) \in \mathcal{P},\ j \in \mathcal{J}\}$. $\mathbb{C}(\mathcal{J}) = \{j \mid x_j \in \mathbb{A}(\mathcal{J})\}$.

Definition 5: The set of affine points defined by $\mathcal{J}$ forms a maximum semi-grid if $|\mathbb{B}_{x_j}(\mathcal{J})| = w, \forall x_j \in \mathbb{A}(\mathcal{J})$.

Example 1: Consider the Hermitian curve $H_2 = x^3 + y^2 + y$ defined over $\mathbb{F}_4 = \{\sigma_0, \sigma_1, \sigma_2, \sigma_3\}$. There are eight affine points on H_2: $P_0 = (\sigma_0, \sigma_0)$, $P_1 = (\sigma_0, \sigma_1)$, $P_2 = (\sigma_1, \sigma_2)$, $P_3 = (\sigma_1, \sigma_3)$, $P_4 = (\sigma_2, \sigma_2)$, $P_5 = (\sigma_2, \sigma_3)$, $P_6 = (\sigma_3, \sigma_2)$, $P_7 = (\sigma_3, \sigma_3)$. Given $\mathcal{J} = \{0, 1, 2\}$, $\mathbb{A}(\mathcal{J}) = \{\sigma_0, \sigma_1\}$, $\mathbb{B}_{\sigma_0}(\mathcal{J}) = \{\sigma_0, \sigma_1\}$ and $\mathbb{B}_{\sigma_1}(\mathcal{J}) = \{\sigma_2\}$. Since $|\mathbb{B}_{\sigma_0}(\mathcal{J})| = 2$ but $|\mathbb{B}_{\sigma_1}(\mathcal{J})| = 1$, the affine points defined by $\mathcal{J}$ do not form a maximum semi-grid. However, the affine points defined by $\mathbb{C}(\mathcal{J}) = \{0, 1, 2, 3\}$ form a maximum semi-grid.

Given an affine point index set $\mathcal{J} \subseteq [0:n-1]$, the Lagrange interpolation polynomial defined by $\mathcal{J}$ is written as

$$L_{\mathcal{J},j}(x,y) = \prod_{\alpha \in \mathbb{A}(\mathcal{J})\backslash\{x_j\}} \frac{x-\alpha}{x_j-\alpha} \prod_{\beta \in \mathbb{B}_{x_j}(\mathcal{J})\backslash\{y_j\}} \frac{y-\beta}{y_j-\beta}. \tag{9}$$

Note that $L_{\mathcal{J},j}(P_j) = 1$ and $L_{\mathcal{J},j}(P_{j'}) = 0$, if $j \neq j'$, where $j, j' \in \mathcal{J}$. Based on the above definitions, we define the following polynomials

$$G(x) = \prod_{\alpha \in \mathbb{A}([0:n-1])} (x-\alpha) \tag{10}$$

and

$$K(x,y) = \sum_{j \in [0:n-1]} \omega_j L_{[0:n-1],j}(x,y), \tag{11}$$

As a result, $K(P_j) = \omega_j$. Note that G and $z-K$ interpolate all points of $\mathbf{P}$ with a multiplicity of one. Let $I_{\mathbf{P},m}$ denote the interpolation module for $\mathbf{P}$. As a module over R, $I_{\mathbf{P},m}$ can be generated by the following $l+1$ polynomials [10]

$$\mathcal{H}^{(\kappa)} = G^{m-\kappa}(z-K)^{\kappa}, \text{if } 0 \leq \kappa \leq m, \tag{12}$$

$$\mathcal{H}^{(\kappa)} = z^{\kappa-m}(z-K)^{m}, \text{if } m < \kappa \leq l. \tag{13}$$

We can also view $I_{\mathbf{P},m}$ as a module that is defined over $\mathbb{F}_q[x]$. It can be generated by the following basis

$$\mathcal{M}_{\mathbf{P}} = \{M^{(s)} \mid M^{(s)} = y^{(s \bmod w)}\mathcal{H}^{(\lfloor \frac{s}{w} \rfloor)}\}, \tag{14}$$

where $0 \leq s \leq lw+w-1$. If $I_{\mathbf{P},m}$ is understood as a module over $\mathbb{F}_q[x]$, the computation of the desired Gröbner basis can be realized efficiently. Given $\mathbf{P}$ and $\underline{\omega}$, polynomials G and K can be defined. They constitute the module generators of (12) and (13), which lead to the module basis construction defined by (14). Afterwards, $\mathcal{M}_{\mathbf{P}}$ will be reduced into a Gröbner basis, in which the minimum polynomial will be chosen as the interpolation polynomial $\mathcal{Q}$.

III. Test-Vectors Formulation and Re-Encoding Transform

This section introduces the test-vectors formulation and the re-encoding transform of the interpolation points.

A. Test-Vectors Formulation

Although the test-vectors formulation in this work is identical to previous LCC decoding schemes, we include it for completeness. Assume that a Hermitian codeword $\underline{c}$ is transmitted and $\underline{\chi} = (\chi_0, \chi_1, \ldots, \chi_{n-1})$ is the channel output. The reliability matrix $\mathbf{\Pi} \in \mathbb{R}^{q \times n}$ with entries $\pi_{i,j} = \Pr(\chi_j \mid c_j = \sigma_i)$ can be obtained, where $0 \leq i \leq q-1$ and $0 \leq j \leq n-1$. Let $i_j^{\mathrm{I}} = \arg\max_i\{\pi_{i,j}\}$ and $i_j^{\mathrm{II}} = \arg\max_{i, i \neq i_j^{\mathrm{I}}}\{\pi_{i,j}\}$. The two most likely decisions of codeword symbol c_j are $\omega_j^{\mathrm{I}} = \sigma_{i_j^{\mathrm{I}}}$ and $\omega_j^{\mathrm{II}} = \sigma_{i_j^{\mathrm{II}}}$. The following likelihood ratio is defined to assess the reliability of χ_j

$$\gamma_j = \frac{\pi_{i_j^{\mathrm{I}},j}}{\pi_{i_j^{\mathrm{II}},j}}, \tag{15}$$

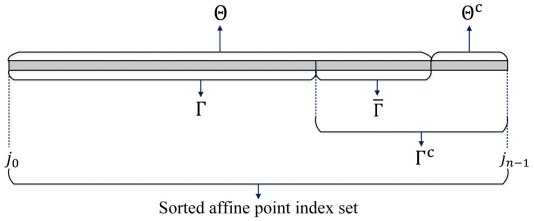

Fig. 1. Index sets of the LCC decoding algorithm.

where $\gamma_j \in [1, \infty)$. A greater γ_j indicates that χ_j is more reliable, and vice versa. By sorting γ_j in a descending order, a new symbol index sequence $j_0, j_1, \ldots, j_{n-1}$ can be obtained. It indicates $\gamma_{j_0} \geq \gamma_{j_1} \geq \cdots \geq \gamma_{j_{n-1}}$. Let $\Theta = \{j_0, j_1, \ldots, j_{n-\eta-1}\}$ denote the index set of the $n-\eta$ most reliable symbols, and its complementary set is $\Theta^{\mathrm{c}} = [0:n-1]\backslash\Theta$. Therefore, all test-vectors can be written as

$$\underline{\omega}_u = (\omega_{j_0}^{(u)}, \omega_{j_1}^{(u)}, \ldots, \omega_{j_{n-1}}^{(u)}), \tag{16}$$

where $u = 0, 1, \ldots, 2^{\eta}-1$, among which $\omega_j^{(u)} = \omega_j^{\mathrm{I}}$, if $j \in \Theta$; and $\omega_j^{(u)} = \omega_j^{\mathrm{I}}$ or ω_j^{II}, if $j \in \Theta^{\mathrm{c}}$. Since there are two decisions for each of the η unreliable symbols, 2^{η} test-vectors can be formulated. This leads to 2^{η} sets of interpolation points that are defined as

$$\mathbf{P}^{(u)} = \{(P_{j_0}, \omega_{j_0}^{(u)}), (P_{j_1}, \omega_{j_1}^{(u)}), \ldots, (P_{j_{n-1}}, \omega_{j_{n-1}}^{(u)})\}. \tag{17}$$

Note that all test-vectors share $n-\eta$ common interpolation points $(P_j, \omega_j^{(u)}), \forall j \in \Theta$. Given $\mathcal{J} \subseteq [0:n-1]$, we use $\mathbf{P}_{\mathcal{J}}^{(u)}$ to denote the subset of interpolation points in $\mathbf{P}^{(u)}$ as $\mathbf{P}_{\mathcal{J}}^{(u)} = \{(P_j, \omega_j^{(u)}) \mid j \in \mathcal{J}\}$. Therefore, $\mathbf{P}^{(u)}$ can be further partitioned into $\mathbf{P}_{\Theta}^{(u)}$ and $\mathbf{P}_{\Theta^{\mathrm{c}}}^{(u)}$.

B. Re-Encoding Transform

We now introduce the re-encoding transform for $\mathbf{P}^{(u)}$. It shifts the received word with a regenerated codeword based on the code's linear property. The shifted received word has some zero entries. The set of interpolation points can be transformed accordingly, leading to a reduced interpolation complexity.

For each test-vector, some interpolation points will be chosen from $\mathbf{P}_{\Theta}^{(u)}$ for the re-encoding. Let $\Gamma \subseteq \Theta$ denote the index set of the re-encoding points. Let $\xi = |\Gamma|$ denote the number of re-encoding points. Let $\Gamma^{\mathrm{c}} = [0:n-1]\backslash\Gamma$ and $\overline{\Gamma} = \Gamma^{\mathrm{c}} \cap \Theta$. For a better illustration, the above mentioned index sets are shown in Fig. 1. With the received word $\underline{\omega}$, the re-encoding polynomial $K_{\Gamma}(x,y) \in \mathcal{L}(\mu P_{\infty})$ needs to be constructed. Based on the Lagrange interpolation polynomial, the re-encoding polynomial K_{Γ} can be defined as

$$K_{\Gamma}(x,y) = \sum_{j \in \Gamma} \omega_j L_{\Gamma,j}(x,y). \tag{18}$$

Lemma 1: The re-encoding polynomial K_{Γ} satisfies $K_{\Gamma}(P_j) = \omega_j, \forall j \in \Gamma$. Furthermore, if $\xi \leq w\lfloor \frac{k-g}{w} \rfloor$ and the affine points defined by Γ form a maximum semi-grid, $K_{\Gamma} \in \mathcal{L}(\mu P_{\infty})$.

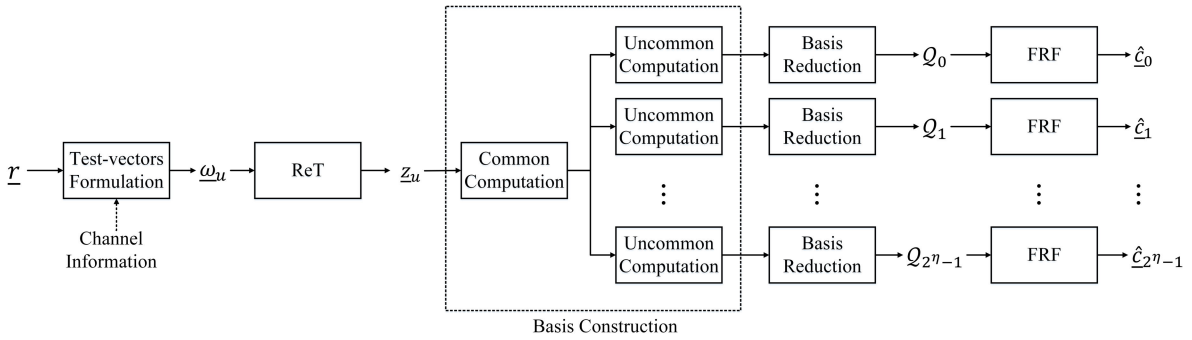

Fig. 2. Block diagram of the proposed LCC decoding algorithm.

Note that the re-encoding using K_Γ can be considered as the erasure decoding using Lagrange interpolation. Based on Lemma 1, Γ will be chosen from Θ for the re-encoding. We further introduce the following definition.

Definition 6: Given an affine point index set $\mathcal{J} \subseteq [0 : n-1]$, $\mathbb{S}(\mathcal{J}) = \{j \mid |\mathbb{B}_{x_j}(\mathcal{J})| = w\}$.

Note that the affine points defined by $\mathbb{S}(\mathcal{J})$ form a maximum semi-grid and $|\mathbb{S}(\mathcal{J})| \leq |\mathcal{J}|$. Based on Lemma 1, if $|\mathbb{S}(\Theta)| \geq w\lfloor\frac{k-g}{w}\rfloor$, $w\lfloor\frac{k-g}{w}\rfloor$ positions will be chosen as Γ such that $\Gamma = \mathbb{S}(\Gamma)$ and $\Gamma \subseteq \Theta$. Otherwise, $\Gamma = \mathbb{S}(\Theta)$. The re-encoding polynomial K_Γ will be constructed with the re-encoding points defined by Γ.

Example 2 (Continued): Consider the $(8, 5)$ Hermitian code defined over $\mathbb{F}_4$. Let $\{j_0, j_1, \ldots, j_7\} = \{4, 6, 0, 3, 5, 7, 1, 2\}$. First, let $\eta = 1$. $\Theta = \{4, 6, 0, 3, 5, 7, 1\}$. $\Theta^c = \{2\}$. We first obtain $\mathbb{S}(\Theta) = \{4, 6, 0, 5, 7, 1\}$. Since $|\mathbb{S}(\Theta)| = 6 > w\lfloor\frac{k-g}{w}\rfloor = 4$, $\Gamma = \{0, 1, 4, 5\}$, $\{0, 1, 6, 7\}$ or $\{4, 5, 6, 7\}$. Secondly, let $\eta = 3$. $\Theta = \{4, 6, 0, 3, 5\}$. $\Theta^c = \{7, 1, 2\}$. We first obtain $\mathbb{S}(\Theta) = \{4, 5\}$. Since $|\mathbb{S}(\Theta)| = 2 < w\lfloor\frac{k-g}{w}\rfloor = 4$, $\Gamma = \mathbb{S}(\Theta) = \{4, 5\}$.

Hence, the re-encoding codeword $\underline{h} = (h_0, h_1, \ldots, h_{n-1}) \in \mathbb{F}_q^n$ can be generated by

$$h_j = K_\Gamma(P_j), \forall j \in [0 : n-1]. \tag{19}$$

Note that $h_j = \omega_j$, if $j \in \Gamma$. Armed with the above knowledge, all Chase decoding test-vectors can be transformed by

$$\underline{\omega}_u \rightarrow \underline{z}_u : z_j^{(u)} = \omega_j^{(u)} - h_j, \forall j \in [0 : n-1]. \tag{20}$$

Note that $z_j^{(u)} = 0$, if $j \in \Gamma$. Consequently, the transformed test-vectors become

$$\underline{z}_u = (z_{j_0}^{(u)}, z_{j_1}^{(u)}, \ldots, z_{j_{n-1}}^{(u)}). \tag{21}$$

This leads to 2^η sets of transformed interpolation points that are defined as

$$\mathbf{P}'_u = \{(P_{j_0}, z_{j_0}^{(u)}), (P_{j_1}, z_{j_1}^{(u)}), \ldots, (P_{j_{n-1}}, z_{j_{n-1}}^{(u)})\}. \tag{22}$$

Note that $\mathbb{S}(\Theta) \cap \mathbb{C}(\Theta^c) = \emptyset$. Since $\Gamma \subseteq \mathbb{S}(\Theta)$, the x-coordinate of points defined by Γ are not contained in $\mathbb{A}(\Theta^c)$. Hence, among all test-vectors there are ξ common points defined by Γ. Their z-coordinates are transformed to zero. The transformed test-vectors still maintain their z-coordinate disparity only in Θ^c. Moreover, the above ReT lifts the restriction on the number of unreliable positions in [21]. This yields a more flexible test-vectors formulation.

IV. The BR Interpolation

This section introduces the BR interpolation for the LCC decoding of Hermitian codes. The low complexity BR interpolation consists of the common computation for all test-vectors and the uncommon computation for each individual one. During the common computation, a partial interpolation module basis is constructed. Afterwards, the uncommon computation further completes the module basis construction for each test-vector and reduces them into their respective Gröbner bases. The desired interpolation polynomials w.r.t. each test-vector can be retrieved from the Gröbner bases. A block diagram of the proposed algorithm is shown in Fig. 2. We begin with introducing the module basis isomorphism.

A. Module Basis Isomorphism

In the LCC decoding, each test-vector is decoded by the GS algorithm with $m = l = 1$. Hence, for test-vector $\underline{\omega}_u$, (12) are simplified into

$$\mathcal{H}_u^{(\kappa)} = G^{1-\kappa}(z - K_u)^\kappa, \tag{23}$$

where

$$K_u = \sum_{j \in [0:n-1]} \omega_j^{(u)} L_j(x, y) \tag{24}$$

and $\kappa = 0$ or 1. It can be used to generate a module for the interpolation points of $\mathbf{P}^{(u)}$. Let $I_{\mathbf{P}^{(u)}}$ denote the interpolation module for $\mathbf{P}^{(u)}$. Based on (14), $I_{\mathbf{P}^{(u)}}$ can be generated by the following module basis

$$\mathcal{M}_{\mathbf{P}^{(u)}} = \{M_u^{(s)} \mid M_u^{(s)} = \mathcal{H}_u^{(\lfloor \frac{s}{w} \rfloor)} y^{(s \bmod w)}, 0 \leq s < 2w\}. \tag{25}$$

Let $I_{\mathbf{P}'_u}$ denote the interpolation module for the transformed points $\mathbf{P}'_u$. The following lemma describes the relationship between the minimum polynomials of $I_{\mathbf{P}^{(u)}}$ and $I_{\mathbf{P}'_u}$.

Lemma 2: $\mathcal{Q}_u$ is the minimum polynomial of $I_{\mathbf{P}^{(u)}}$ iff $\mathcal{Q}'_u(x, y, z) = \mathcal{Q}_u(x, y, z + K_\Gamma)$ is also the one of $I_{\mathbf{P}'_u}$.

Therefore, the minimum polynomial $\mathcal{Q}_u$ of $I_{\mathbf{P}^{(u)}}$ can be obtained by $\mathcal{Q}_u(x, y, z) = \mathcal{Q}'_u(x, y, z - K_\Gamma)$. Note that polynomial G can be written as

$$G = G_\Gamma G_{\Gamma^c}, \tag{26}$$

where

$$G_{\Gamma}(x) = \prod_{\alpha \in \mathbb{A}(\Gamma)} (x - \alpha) \tag{27}$$

and

$$G_{\Gamma^{\mathrm{c}}}(x) = \prod_{\alpha \in \mathbb{A}(\Gamma^{\mathrm{c}})} (x - \alpha). \tag{28}$$

Based on the above mentioned test-vectors formulation, among the $n-\eta$ common interpolation points shared by the transformed test-vectors, the z-coordinates of ξ re-encoding points are set to zero, i.e., $(P_j, 0), \forall j \in \Gamma$. Therefore, based on $\underline{z}_u$ and $\mathbf{P}'_u$, K'_u can be defined as

$$\begin{aligned} K'_u &= \sum_{j \in [0:n-1]} z_j^{(u)} \cdot L_j \\ &= \sum_{j \in \Gamma} 0 \cdot L_j + \sum_{j \in \Gamma^{\mathrm{c}}} z_j^{(u)} \cdot L_j \\ &= \sum_{j \in \Gamma^{\mathrm{c}}} z_j^{(u)} \cdot \prod_{\alpha \in \mathbb{A}(J) \setminus \{x_j\}} \frac{x - \alpha}{x_j - \alpha} \cdot \prod_{\beta \in \mathbb{B}_{x_j}(J) \setminus \{y_j\}} \frac{y - \beta}{y_j - \beta} \\ &= \sum_{j \in \Gamma^{\mathrm{c}}} z_j^{(u)} \cdot \prod_{\alpha \in \mathbb{A}(\Gamma)} \frac{x - \alpha}{x_j - \alpha} \cdot L_{\Gamma^{\mathrm{c}},j} \\ &= G_{\Gamma} \cdot \sum_{j \in \Gamma^{\mathrm{c}}} \frac{z_j^{(u)}}{G_{\Gamma}(x_j)} \cdot L_{\Gamma^{\mathrm{c}},j}. \end{aligned} \tag{29}$$

Polynomials G and K'_u interpolate the points of $\mathbf{P}'_u$ with a multiplicity of one. Based on (26) and (29), G_{Γ} becomes the common factor of G and K'_u. Hence, entries of the module basis $\mathcal{M}_{\mathbf{P}^{(u)}}$ can be further transformed, resulting in a module basis isomorphism. Let

$$\widetilde{z}_j^{(u)} = \frac{z_j^{(u)}}{G_{\Gamma}(x_j)}, \forall j \in \Gamma^{\mathrm{c}}. \tag{30}$$

For the set of transformed interpolation points $\mathbf{P}'_u$, its subset $\{(P_j, z_j^{(u)}) \mid j \in \Gamma^{\mathrm{c}}\}$ can be further transformed into

$$\widetilde{\mathbf{P}}_u = \{(P_j, \widetilde{z}_j) \mid j \in \Gamma^{\mathrm{c}}\}. \tag{31}$$

Let $I_{\widetilde{\mathbf{P}}_u}$ denote the interpolation module for $\widetilde{\mathbf{P}}_u$. The following bijective mapping is an $\mathbb{F}_q[x]$-module isomorphism between $I_{\mathbf{P}'_u}$ and $I_{\widetilde{\mathbf{P}}_u}$

$$\begin{aligned} &\vartheta : I_{\mathbf{P}'_u} \to I_{\widetilde{\mathbf{P}}_u} \\ &Q'_u(x, y, z) = G_{\Gamma} \widetilde{Q}_u(x, y, \frac{z}{G_{\Gamma}}) \to \widetilde{Q}_u(x, y, z). \end{aligned} \tag{32}$$

Let us define

$$\mathcal{K}_{\Gamma^{\mathrm{c}}}^{(u)} = \sum_{j \in \Gamma^{\mathrm{c}}} \widetilde{z}_j^{(u)} L_{\Gamma^{\mathrm{c}},j}. \tag{33}$$

The module basis of $I_{\widetilde{\mathbf{P}}_u}$ can be defined as

$$\mathcal{M}_{\widetilde{\mathbf{P}}_u} = \{\widetilde{M}_u^{(s)} \mid \widetilde{M}_u^{(s)} = \widetilde{\mathcal{H}}_u^{(\lfloor \frac{s}{w} \rfloor)} y^{(s \bmod w)}\}, \tag{34}$$

where

$$\widetilde{\mathcal{H}}_u^{(\kappa)} = G_{\Gamma^{\mathrm{c}}}^{1-\kappa} (z - \mathcal{K}_{\Gamma^{\mathrm{c}}}^{(u)})^{\kappa}, \tag{35}$$

$\kappa = 0$ or 1, and $0 \le s < 2w$.

Lemma 3: $\mathcal{M}_{\widetilde{\mathbf{P}}_u}$ is a basis for the interpolation module $I_{\widetilde{\mathbf{P}}_u}$.

Based on (35), the entries of $\mathcal{M}_{\widetilde{\mathbf{P}}_u}$ are

$$\widetilde{M}_u^{(s)} = G_{\Gamma^{\mathrm{c}}} y^s, 0 \le s < w, \tag{36}$$

$$\widetilde{M}_u^{(s)} = (z - \mathcal{K}_{\Gamma^{\mathrm{c}}}^{(u)}) y^{(s \bmod w)}, w \le s < 2w. \tag{37}$$

Lemma 4: Assume that $\mathcal{Q}_u$ and $\widetilde{\mathcal{Q}}_u$ are the minimum polynomial of $I_{\mathbf{P}^{(u)}}$ and $I_{\widetilde{\mathbf{P}}_u}$, respectively, and $\widetilde{w}_z = w_z - \xi$. Their weighted degrees satisfy $\deg_{1,w_z} \mathcal{Q}_u = \deg_{1,\widetilde{w}_z} G_{\Gamma} + \deg_{1,\widetilde{w}_z} \widetilde{\mathcal{Q}}_u$.

Lemma 4 implies that polynomials in $I_{\widetilde{\mathbf{P}}_u}$ are organized under the $(1, \widetilde{w}_z)$-revlex order. The relationship between the minimum polynomials of $I_{\mathbf{P}^{(u)}}$ and $I_{\widetilde{\mathbf{P}}_u}$ is described as follows.

Lemma 5: Given $\widetilde{\mathcal{Q}}_u$ as the minimum polynomial of $I_{\widetilde{\mathbf{P}}_u}$, the minimum polynomial $\mathcal{Q}_u$ of $I_{\mathbf{P}^{(u)}}$ can be obtained by

$$\mathcal{Q}_u(x, y, z) = G_{\Gamma}(x) \widetilde{\mathcal{Q}}_u(x, y, \frac{z - K_{\Gamma}}{G_{\Gamma}}). \tag{38}$$

Therefore, with $\widetilde{\mathbf{P}}_u$, the isomorphic module basis $\mathcal{M}_{\widetilde{\mathbf{P}}_u}$ can be constructed. It is further reduced into the desired Gröbner basis which contains $\widetilde{\mathcal{Q}}_u$. It can then be restored into the desired interpolation polynomial $\mathcal{Q}_u$. Be aware that $\widetilde{\mathcal{H}}_u^{(\kappa)}$ has lower x-degree than $\mathcal{H}_u^{(\kappa)}$, which results in a reduced basis reduction complexity. The above procedures can be categorized into the common computation (in basis reduction) and the remaining uncommon computation (in basis construction and their reduction), which are described in the following subsections.

B. Common Computation

Based on (36) and (37), in all the module bases $\mathcal{M}_{\widetilde{\mathbf{P}}_u}$, $\widetilde{M}_u^{(0)}$, $\widetilde{M}_u^{(1)}$, …, $\widetilde{M}_u^{(w-1)}$ are the common entries, while $\widetilde{M}_u^{(w)}$, $\widetilde{M}_u^{(w+1)}$, …, $\widetilde{M}_u^{(2w-1)}$ are the uncommon ones. Therefore, $G_{\Gamma^{\mathrm{c}}}$ can be computed once and shared by the decoding of all test-vectors. Since

$$\mathcal{K}_{\Gamma^{\mathrm{c}}}^{(u)} = \mathcal{K}_{\overline{\Gamma}} + \mathcal{K}_{\Theta^{\mathrm{c}}}^{(u)}, \tag{39}$$

where

$$\mathcal{K}_{\overline{\Gamma}} = \sum_{j \in \overline{\Gamma}} \widetilde{z}_j^{(u)} L_{\Gamma^{\mathrm{c}},j} \tag{40}$$

and

$$\mathcal{K}_{\Theta^{\mathrm{c}}}^{(u)} = \sum_{j \in \Theta^{\mathrm{c}}} \widetilde{z}_j^{(u)} L_{\Gamma^{\mathrm{c}},j}, \tag{41}$$

polynomials $\widetilde{M}_u^{(s)}$, where $w \le s < 2w$, can be rewritten as

$$\widetilde{M}_u^{(s)} = (z - \mathcal{K}_{\overline{\Gamma}} - \mathcal{K}_{\Theta^{\mathrm{c}}}^{(u)}) y^{(s \bmod w)}. \tag{42}$$

Hence, in constructing the 2^{η} module bases, $\mathcal{K}_{\overline{\Gamma}}$ can also be computed once and shared by the decoding of 2^{η} test-vectors.

C. Uncommon Computation

The uncommon computation further completes the construction of the module basis for each individual transformed test-vector, and subsequently reduces them into the desired Gröbner bases. Based on (42), in order to construct the module basis for different test-vectors, polynomial $\mathcal{K}_{\Theta^c}^{(u)}$ needs to be further computed. Hence, the isomorphic module basis $\mathcal{M}_{\widetilde{\mathbf{P}}_u}$ w.r.t. each transformed test-vector can be formed as in (34). Afterwards, the Mulders-Storjohann (MS) algorithm [32] will be applied to reduce $\mathcal{M}_{\widetilde{\mathbf{P}}_u}$ into the desired Gröbner basis which contains the minimum polynomial $\widetilde{\mathcal{Q}}_u$ of $I_{\widetilde{\mathbf{P}}_u}$.

Since $\widetilde{\mathcal{Q}}_u$ can be written as

$$\widetilde{\mathcal{Q}}_u(x,y,z) = \widetilde{\mathcal{Q}}_u^{(0)}(x,y) + \widetilde{\mathcal{Q}}_u^{(1)}(x,y)\cdot z, \tag{43}$$

which interpolates all points of $\widetilde{\mathbf{P}}_u$, based on (38), it can be restored into $\mathcal{Q}_u$ by

$$\mathcal{Q}_u(x,y,z) = G_\Gamma(x)\widetilde{\mathcal{Q}}_u^{(0)}(x,y) + \widetilde{\mathcal{Q}}_u^{(1)}(x,y)\cdot(z - K_\Gamma). \tag{44}$$

It interpolates all points of $\mathbf{P}^{(u)}$.

The ReT-assisted BR interpolation for LCC decoding is summarized as in Algorithm 1. Since the uncommon computation for different test-vectors can be performed in a fully parallel manner, it not only results in a low decoding complexity, but also a low decoding latency.

Algorithm 1 The ReT-Assisted BR Interpolation

Input: $\mathbf{\Pi}$, $\underline{\omega}$;
Output: $\mathcal{Q}_u$;

1: Formulate 2^η test-vectors $\underline{\omega}_u$ as in (16);
2: Perform the re-encoding and transform $\underline{\omega}_u$ yielding $\underline{z}_u$ as in (20);
3: Construct the common computation in basis construction as in (36), (37) and (40);
4: **for** $u = 0$ to $2^\eta - 1$ **do**
5: Complete the construction of $\mathcal{M}_{\widetilde{\mathbf{P}}_u}$ for each test-vector as in (37) and (41);
6: Perform the MS algorithm, reducing $\mathcal{M}_{\widetilde{\mathbf{P}}_u}$ into a Gröbner basis;
7: Identify the minimum polynomial in the reduced $\mathcal{M}_{\widetilde{\mathbf{P}}_u}$ as $\widetilde{\mathcal{Q}}_u$;
8: Restore $\widetilde{\mathcal{Q}}_u$ into $\mathcal{Q}_u$.
9: **end for**

V. Fast Root-Finding

In LCC decoding, the root-finding needs to be performed for 2^η decoding events. Its complexity becomes remarkably high as η increases. Therefore, it is important to reduce the complexity of root-finding. This section proposes the FRF for the LCC decoding. It computes the estimated codewords based on the interpolation outcomes, and hence eliminates the redundant re-encoding computation that is required for identifying the most likely codeword candidate.

Recall that $\mathcal{Q}_u$ is the interpolation polynomial that interpolates all points of $\mathbf{P}^{(u)}$. Let

$$T_u(x,y) = G_\Gamma(x)\widetilde{\mathcal{Q}}_u^{(0)}(x,y). \tag{45}$$

Suppose that there exists a polynomial $\widetilde{f}_u(x,y) \in \mathcal{L}(\mu P_\infty)$ satisfying

$$T_u + \widetilde{\mathcal{Q}}_u^{(1)}\cdot\widetilde{f}_u = 0, \tag{46}$$

it can be determined as

$$\widetilde{f}_u = -\frac{T_u}{\widetilde{\mathcal{Q}}_u^{(1)}}. \tag{47}$$

The estimated message polynomial $\hat{f}_u$ can be further determined as

$$\hat{f}_u = \widetilde{f}_u + K_\Gamma. \tag{48}$$

Let $\underline{\hat{c}}_u = (\hat{c}_0^{(u)}, \hat{c}_1^{(u)}, \ldots, \hat{c}_{n-1}^{(u)})$ and $\underline{\widetilde{c}}_u = (\widetilde{c}_0^{(u)}, \widetilde{c}_1^{(u)}, \ldots, \widetilde{c}_{n-1}^{(u)})$ denote the corresponding codewords of $\hat{f}_u$ and $\widetilde{f}_u$, respectively, which are obtained in the encoding fashion of (5). Hence, the estimated codeword symbol $\hat{c}_j^{(u)}$ can be determined as

$$\begin{aligned}\hat{c}_j^{(u)} &= \hat{f}_u(P_j)\\ &= \widetilde{f}_u(P_j) + K_\Gamma(P_j)\\ &= \widetilde{c}_j^{(u)} + h_j.\end{aligned} \tag{49}$$

Therefore, based on (47) and (49), the estimated codeword $\underline{\hat{c}}_u$ can be computed directly from the interpolation outcomes. Within the Chase decoding output list, the most likely codeword candidate will be chosen, which is denoted as $\underline{\hat{c}} = (\hat{c}_0, \hat{c}_1, \ldots, \hat{c}_{n-1})$. Note that the decoded message polynomial, denoted by $\hat{f}(x,y)$, can be obtained by unencoding $\underline{\hat{c}}$ [33]. The proposed FRF is further built upon the realization of (47), which needs to be categorized into the following five cases.

Case I: Based on (46) and (48),

$$T_u(P_j) + \widetilde{\mathcal{Q}}_u^{(1)}(P_j)\cdot(\hat{c}_j^{(u)} - h_j) = 0, \forall j \in [0:n-1]. \tag{50}$$

Since $\mathcal{Q}_u$ interpolates all points of $\mathbf{P}^{(u)}$,

$$T_u(P_j) + \widetilde{\mathcal{Q}}_u^{(1)}(P_j)\cdot(\omega_j^{(u)} - h_j) = 0, \forall j \in [0:n-1]. \tag{51}$$

Therefore, if $\widetilde{\mathcal{Q}}_u^{(1)}(P_j) \neq 0$, $\hat{c}_j^{(u)} = \omega_j^{(u)}$, which indicates that $\omega_j^{(u)}$ is correct.

Case II: If $T_u(P_j) = \widetilde{\mathcal{Q}}_u^{(1)}(P_j) = 0$, the partial derivatives of T_u and $\widetilde{\mathcal{Q}}_u^{(1)}$ are needed such that

$$\widetilde{f}_u(P_j) = -\frac{\partial T_u}{\partial x}\Big/\frac{\partial \widetilde{\mathcal{Q}}_u^{(1)}}{\partial x}\Big|_{P_j}. \tag{52}$$

Note that for the Hermitian curve H_w defined in (1),

$$\frac{\partial x^{w+1}}{\partial x} = \frac{\partial y^w}{\partial x} + \frac{\partial y}{\partial x}. \tag{53}$$

Since $q = w^2$,

$$\frac{\partial x^{w+1}}{\partial x} = (w+1)\cdot x^w = x^w \tag{54}$$

and

$$\frac{\partial y^w}{\partial x} = w \cdot y^{w-1} \cdot \frac{\partial y}{\partial x} = 0. \tag{55}$$

Therefore, (53) can be written as

$$\frac{\partial y}{\partial x} = x^w. \tag{56}$$

Consequently, (52) can be calculated with the assistance of (56).

If both $T_u(P_j) = \widetilde{\mathcal{Q}}_u^{(1)}(P_j) = 0$ and $\frac{\partial T_u}{\partial x}|_{P_j} = \frac{\partial \widetilde{\mathcal{Q}}_u^{(1)}}{\partial x}|_{P_j} = 0$, the estimated codeword symbols cannot be determined by **Cases I** and **II**. Their estimations need to be realized through either the erasure decoding, or the RCS process. Let $\Lambda_u = \{j \mid T_u(P_j) = \widetilde{\mathcal{Q}}_1^{(u)}(P_j) = \frac{\partial T_u}{\partial x}|_{P_j} = \frac{\partial \widetilde{\mathcal{Q}}_u^{(1)}}{\partial x}|_{P_j} = 0\}$. Let $\mathcal{D}_u = \mathbb{S}([0 : n-1] \backslash \mathbb{C}(\Lambda_u))$. The following lemma describes the erasure decoding that is used to recover the symbols associated with Λ_u.

Lemma 6: Given $\mathcal{D}_u$ and $\{\widetilde{c}_j^{(u)} \mid \forall j \in \mathcal{D}_u\}$, let

$$\mathcal{K}_{\mathcal{D}_u}(x, y) = \sum_{j \in \mathcal{D}_u} \widetilde{c}_j^{(u)} L_{\mathcal{D}_u, j}(x, y). \tag{57}$$

It satisfies $\widetilde{c}_j^{(u)} = \mathcal{K}_{\mathcal{D}_u}(P_j), \forall j \in \mathcal{D}_u$. If $|\mathcal{D}_u| \geq w\lceil \frac{k+g}{w} \rceil$, the symbols associated with Λ_u can be recovered by

$$\widetilde{c}_j^{(u)} = \mathcal{K}_{\mathcal{D}_u}(P_j), \forall j \in \Lambda_u. \tag{58}$$

Case III: Based on Lemma 6, if $|\mathcal{D}_u| \geq w\lceil \frac{k+g}{w} \rceil$, the symbols associated with Λ_u will be recovered using the erasure decoding as in (57) and (58).

Case IV: If $|\mathcal{D}_u| < w\lceil \frac{k+g}{w} \rceil$, the symbols associated with Λ_u cannot be recovered using the above erasure decoding. As a result, the RCS algorithm remains the only solution.

Case V: If $T_u(P_j) \neq 0$ but $\widetilde{\mathcal{Q}}_u^{(1)}(P_j) = 0$, or $\frac{\partial T_u}{\partial x}|_{P_j} \neq 0$ but $\frac{\partial \widetilde{\mathcal{Q}}_u^{(1)}}{\partial x}|_{P_j} = 0$, the number of errors in this test-vector exceeds the decoding capability. In this case, root-finding for this test-vector will be terminated.

After computing $\widetilde{c}_j^{(u)}$, the estimated codeword symbol can be obtained as in (49). The most likely codeword candidate $\hat{\underline{c}}$ will be chosen from the decoding output list. Note that the unencoding can be performed to further retrieve the message polynomial as [33]

$$\hat{f}(x, y) = \sum_{j \in [0:n-1]} \hat{c}_j L_j(x, y). \tag{59}$$

Unlike the RCS algorithm [5], [6], [7] that estimates the message symbols in a successive manner, the FRF can estimate all codeword symbols in parallel. Afterwards, the message that corresponds to the most likely codeword will be chosen as the decoding output. It should be noted that in case of using the RCS algorithm, all estimated messages need to be encoded. So far, the one that produces the most likely codeword will be identified as the decoding output. The proposed FRF avoids this expensive encoding operation. Note that the proposed FRF and the FF in [23] differ in **Case I**. Besides $\widetilde{\mathcal{Q}}_u^{(1)}$, the FF should also compute the evaluation of T_u. Therefore, the FRF yields a slightly lower complexity than the FF. The FRF is summarized as in Algorithm 2.

Algorithm 2 The FRF Algorithm

Input: G_Γ, $\widetilde{\mathcal{Q}}^{(u)}$, $\underline{h}$;
Output: $\hat{\underline{c}}$, $\hat{f}$;

for $u = 0$ to $2^\eta - 1$ **do**
  Calculate T_u as in (45);
  for $j = 0$ to $n - 1$ **do**
    Case I: $\hat{c}_j^{(u)} = \omega_j^{(u)}$;
    Case II: Determine $\widetilde{c}_j^{(u)}$ as in (52);
    Case III: Recover the symbols as in (57) and (58);
    Case IV: Perform the RCS algorithm;
    Case V: Terminate root-finding for this test-vector;
  end for
  Obtain $\hat{\underline{c}}_u$ as in (49);
end for
Choose the most likely codeword candidate $\hat{\underline{c}}$ and obtain $\hat{f}$ as in (59).

VI. Decoding With Early Termination

Equipped with the above proposed BR interpolation and FRF, the LCC decoding can process the 2^η test-vectors in a parallel manner, resulting in a low decoding latency. However, its decoding complexity remains challengingly high for practice. This section proposes two early termination strategies for the decoding. Consequently, the decoding complexity can be reduced by eliminating the redundant decoding computation. The decoding will be terminated once a codeword that satisfies the ML criterion [24], [25] is found. It has been noticed that as the channel condition improves, such a codeword can be found at an earlier decoding stage. We show that the ML criterion can be used to assess the re-encoding codeword. Once it satisfies the ML criterion, the decoding will be terminated. If all test-vectors are decoded in a sequential manner, we show that they can be decoded progressively in an order supported by their reliabilities. During the progressive decoding, module basis update is needed. Once an ML codeword is found, the decoding will also be terminated.

A. Early Termination Through Re-Encoding

Based on Section III-B, a re-encoding codeword h is generated as in (19). A straightforward method is to use the ML criterion to assess h. If h is an ML codeword, the decoding will be terminated. Note that if the Hamming distance between h and ω is smaller, it is more likely that h satisfies the ML criterion. However, since at most $w\lfloor \frac{k-g}{w} \rfloor$ points are chosen for generating h, it is only guaranteed that h and $\underline{\omega}$ have $w\lfloor \frac{k-g}{w} \rfloor$ identical symbols. Hence, h can rarely satisfy the ML criterion. As mentioned in Section III-B, the re-encoding can be considered as the erasure decoding. As the channel condition improves, $\underline{\omega}$ becomes less corrupted. Based on Lemma 6, the erasure decoding that is similar to (57) and (58) can be used to generate a re-encoding codeword. It should have a smaller Hamming distance to $\underline{\omega}$ under this condition. Let $\mathcal{T}$ denote the affine point index set for generating such a codeword. In order to ensure that the erasure decoding

succeeds, $\mathcal{T}$ shall include more reliable symbol positions. Let us initialize $\mathcal{T}$ as $\mathcal{T} = \mathbb{S}([0:n-1])$, and $\tilde{j}$ denote the least reliable position in $\mathcal{T}$. $\mathcal{T}$ shall be further updated as

$$\mathcal{T} = \mathcal{T} \backslash \mathbb{C}(\tilde{j}). \tag{60}$$

This above position exclusion continues until $|\mathcal{T}| \leq w\lceil \frac{k+g}{w} \rceil$.

Note that the erasure decoding based on $\mathcal{T}$ does not always yield an (n,k) Hermitian codeword, let alone a codeword that satisfies the ML criterion. Therefore, after updating $\mathcal{T}$, another affine point index set Γ' will continue to be generated. It needs to satisfy the same constraints as Γ of Lemma 1. Let us initialize Γ' as $\Gamma' = \mathcal{T}$, and j' denote the least reliable position in Γ'. Γ' shall be updated as

$$\Gamma' = \Gamma' \backslash \mathbb{C}(j'). \tag{61}$$

This above position exclusion of Γ' continues until $|\Gamma'| \leq w\lfloor \frac{k-g}{w} \rfloor$ and $\Gamma' \cap \mathbb{C}(\Theta^c) = \emptyset$. If an (n,k) Hermitian codeword is not yielded based on $\mathcal{T}$, Γ' will be utilized to generate a re-encoding codeword and transform $\mathbf{P}^{(u)}$ in the same way as Γ in Section III-B.

Example 3 (Continued): In decoding the $(8,5)$ Hermitian code mentioned in Example 2, $\eta = 3$, $\Theta = \{4,6,0,3,5\}$ and $\Theta^c = \{7,1,2\}$. Let $\mathcal{T} = \{0,1,2,3,4,5,6,7\}$. It is updated into $\{0,1,4,5,6,7\}$. Let $\Gamma' = \{0,1,4,5,6,7\}$. It is further updated into $\{4,5\}$.

Based on $\mathcal{T}$ and Γ', the re-encoding polynomials are defined as

$$K_{\mathcal{T}}(x,y) = \sum_{j \in \mathcal{T}} \omega_j L_{\mathcal{T},j}(x,y) \tag{62}$$

and

$$K_{\Gamma'}(x,y) = \sum_{j \in \Gamma'} \omega_j L_{\Gamma',j}(x,y), \tag{63}$$

respectively. As in (62) and (63), $L_{\mathcal{T},j}$ and $L_{\Gamma',j}$ should be computed. The above description shows that $\Gamma' \subseteq \mathcal{T}$. Hence, $L_{\mathcal{T},j}$ can be computed upon $L_{\Gamma',j}$, indicating that $K_{\mathcal{T}}$ and $K_{\Gamma'}$ can be derived by sharing common computations. In particular, if $j \in \Gamma'$,

$$L_{\mathcal{T},j} = L_{\Gamma',j} \cdot \prod_{\alpha \in \mathbb{A}(\mathcal{T} \backslash \Gamma')} \frac{x-\alpha}{x_j - \alpha}. \tag{64}$$

After constructing $K_{\mathcal{T}}$, it is necessary to check if $K_{\mathcal{T}} \in \mathcal{L}(\mu P_\infty)$.
If so, a Hermitian codeword $\underline{\tilde{h}} = (\tilde{h}_0, \tilde{h}_1, \ldots, \tilde{h}_{n-1})$ will be generated by

$$\tilde{h}_j = K_{\mathcal{T}}(P_j), \forall j \in [0:n-1]. \tag{65}$$

If $\underline{\tilde{h}}$ satisfies the ML criterion, the decoding will be terminated. If $K_{\mathcal{T}} \notin \mathcal{L}(\mu P_\infty)$, or $\underline{\tilde{h}}$ cannot satisfy the ML criterion, another Hermitian codeword $\underline{h}'$ will be generated by

$$h'_j = K_{\Gamma'}(P_j), \forall j \in [0:n-1]. \tag{66}$$

It can still play a similar role as $\underline{h}$ does for the ReT.

It is observed that if the received symbols associated with $\mathcal{T}$ are more reliable, it is more likely that $K_{\mathcal{T}}$ can generate an (n,k) Hermitian codeword. Therefore, the reliability $\pi_{i^{\mathrm{I}}_j,j}$ can be utilized to assess the received symbols associated with $\mathcal{T}$ in determining whether $K_{\mathcal{T}}$ should be computed. Let π_{thrd} denote the reliability threshold. If $\pi_{i^{\mathrm{I}}_j,j} \geq \pi_{\mathrm{thrd}}, \forall j \in \mathcal{T}$, $K_{\mathcal{T}}$ will be computed. Otherwise, only $K_{\Gamma'}$ will be computed for the ReT. We will show that by carefully choosing π_{thrd}, the computation of $K_{\mathcal{T}}$ can only be attempted if $\mathcal{T}$ is reliable enough, avoiding redundant computation.

Remark 1: The re-encoding described in this section can replace the one described in Section III-B. However, the erasure decoding operations of (62) and (65) cannot always yield an (n,k) Hermitian codeword. If the erasure decoding fails, this computation becomes redundant. It can also be aware that although the early termination through re-encoding can reduce the decoding latency, the cost associated with computing $K_{\mathcal{T}}$ and $\underline{\tilde{h}}$ is considerable. Therefore, this re-encoding method is only proposed as an alternative to that of Section III-B, if the early termination of the LCC decoding is desirable.

B. Early Termination Through Ordering the Test-Vectors

Given a test-vector $\underline{\omega}_u = (\omega_0^{(u)}, \omega_1^{(u)}, \ldots, \omega_{n-1}^{(u)})$, its reliability can be determined by

$$\Omega_u = \prod_{j=0}^{n-1} \pi_{i_j^{(u)},j}, \tag{67}$$

where $i_j^{(u)} = \mathrm{index}\{\sigma_i \mid \sigma_i = \omega_j^{(u)}\}$. A greater Ω_u indicates $\underline{\omega}_u$ is more reliable. It is considered to be more likely to yield the intended message. It should be decoded earlier. Since all test-vectors share the common symbols $\omega_j^{(u)}$ with $j \in \Theta$, their reliabilities can be compared by only assessing

$$\hat{\Omega}_u = \prod_{j \in \Theta^c} \pi_{i_j^{(u)},j}. \tag{68}$$

Based on $\hat{\Omega}_u$, all test-vectors are ordered, yielding $\underline{\omega}_{u_0}, \underline{\omega}_{u_1}, \ldots, \underline{\omega}_{u_{2^\eta-1}}$, where $\hat{\Omega}_{u_0} \geq \hat{\Omega}_{u_1} \geq \cdots \geq \hat{\Omega}_{u_{2^\eta-1}}$. The decoding will process the test-vectors in the above order, leading to the progressive decoding [26], [27]. Once an ML codeword is found, the decoding will be terminated.

In this progressive LCC decoding, the module basis of $\underline{\omega}_{u_0}$ will be first constructed. That says instead of computing (40), $\mathcal{K}_{\Gamma^c}^{(u_0)}$ as in (33) will be computed directly. If the decoding of $\underline{\omega}_{u_0}$ cannot produce an ML codeword, $\underline{\omega}_{u_1}$ will be decoded. It can be observed that for the 2^η module bases $\mathcal{M}_{\widetilde{\mathbf{P}}_u}$ of the test-vectors, $\widetilde{M}_u^{(0)}$, $\widetilde{M}_u^{(1)}$, …, $\widetilde{M}_u^{(w-1)}$ are common, while $\widetilde{M}_u^{(w)}$, $\widetilde{M}_u^{(w+1)}$, …, $\widetilde{M}_u^{(2w-1)}$ are uniquely determined by test-vector $\underline{\omega}_u$. Let $\Xi_{u_\zeta} = \{j \mid \omega_j^{(u_\zeta)} \neq \omega_j^{(u_{\zeta+1})}, j \in \Theta^c\}$ denote the index set of different symbols between $\underline{\omega}_{u_\zeta}$ and $\underline{\omega}_{u_{\zeta+1}}$. It can be used to formulate polynomial W_{u_ζ} as

$$W_{u_\zeta} = \sum_{j \in \Xi_{u_\zeta}} (\widetilde{z}_j^{(u_\zeta)} - \widetilde{z}_j^{(u_{\zeta+1})}) L_{\Gamma^c,j}. \tag{69}$$

Based on (69), after decoding test-vector $\underline{\omega}_{u_\zeta}$, $\mathcal{K}_{\Gamma^c}^{(u_{\zeta+1})}$ can be updated as

$$\mathcal{K}_{\Gamma^c}^{(u_{\zeta+1})} = \mathcal{K}_{\Gamma^c}^{(u_\zeta)} + W_{u_\zeta}. \tag{70}$$

Therefore, $\mathcal{M}_{\widetilde{\mathbf{P}}_{u_{\zeta+1}}}$ can be generated based on $\mathcal{M}_{\widetilde{\mathbf{P}}_{u_\zeta}}$ and W_{u_ζ}. The progressive decoding process will be conducted sequentially. It will be terminated once an ML codeword is found. It has been observed that the codeword that satisfies the ML criterion can often be identified by decoding the first few test-vectors, especially when the channel condition is good. This leads to a more significant complexity reduction.

VII. Decoding Complexity and Latency

This section analyses the decoding complexity and latency. The decoding complexity is measured as the average number of finite field multiplications in decoding a codeword. The decoding latency is measured as the average simulation time required in decoding a codeword. First, the theoretical characterization of the complexity is analysed. We focus on the BR interpolation complexity reduction effect brought by the ReT, as well as the complexity advantage of the proposed FRF over the existing RCS algorithm.

We first consider the effect of ReT in reducing the BR interpolation complexity. The common computation in basis construction needs to compute G_{Γ^c} and $\mathcal{K}_{\overline{\Gamma}}$, where computing the latter one dominates the complexity. Since $\deg_x L_{\Gamma^c,j} \leq \frac{n-\xi}{w} - 1$ and $\deg_y L_{\Gamma^c,j} \leq w-1$, the asymptotic complexity of computing $\mathcal{K}_{\overline{\Gamma}}$ is $O((n-\xi)^2)$. Similar to the common computation, the uncommon computation needs to compute $\mathcal{K}^{(u)}_{\Theta^c}$ for each test-vector. Since there are 2^η test-vectors, the asymptotic complexity of computing $\mathcal{K}^{(u)}_{\Theta^c}$ is $O(2^\eta \eta(n-\xi))$. Without the ReT, the asymptotic complexities of common computation and uncommon computation in basis construction are $O(n(n-\eta))$ and $O(2^\eta \eta n)$, respectively. Therefore, it can be seen that the ReT helps reduce the basis construction complexity by a factor of $\frac{\xi}{n}$.

The basis reduction complexity is mainly determined by the maximum polynomial in each module basis. Without the ReT, the maximum polynomial is $y^{w-1}K_u$. Its weighted degree is $n + 2w^2 - w - 2$. The asymptotic complexity of basis reduction is $O(2^\eta n^2)$. With the ReT, polynomial $y^{w-1}\mathcal{K}^{(u)}_{\Gamma^c}$ has the maximum weighted degree. Its weighted degree is $n + 2w^2 - w - 2 - \xi$. The asymptotic complexity of basis reduction is $O(2^\eta (n-\xi)^2)$. Therefore, the ReT can also reduce the basis reduction complexity.

For the FRF, our simulations show that **Cases I** and **II** are the major FRF events. Their computations mainly consist of polynomial evaluations, which can be performed in parallel. This feature will be welcomed by practical implementation. Their asymptotic complexities are $O(2^\eta \mathcal{E} n)$ and $O(2^\eta \mathcal{E}(n-\xi))$, respectively, where $\mathcal{E}$ denotes the number of errors. For the RCS algorithm [5], [6], [7], it exhibits an asymptotic complexity of $O(2^\eta k(n-\xi))$. Note that the Chase decoding is often applied to decode high-rate codes. For high-rate codes, if the decoding can produce a valid output codeword, there is $\mathcal{E} < k$. Therefore, the FRF has a complexity advantage over the RCS algorithm. Please note that in the LCC decoding, the estimated codewords should be generated, so that the most likely codeword candidate and its message can be identified as the decoding output. For the RCS algorithm, it needs to encode the 2^η estimated messages to generate the estimated codewords, exhibiting an asymptotic complexity of $O(2^\eta kn)$. The FRF can compute the estimated codewords directly from the interpolation outcomes. It can avoid this expensive encoding operation. After identifying the most likely codeword candidate, only one unencoding computation will be performed to retrieve the estimated message, exhibiting an asymptotic complexity of $O(n^2)$.

TABLE I
Complexity in Decoding of the $(64, 39)$ Hermitian Code

Alg.	LCC (BR-RCS)		LCC (ReT-Kötter-RCS)		LCC (ReT-BR-FRF)	
	$\eta=3$	$\eta=6$	$\eta=3$	$\eta=6$	$\eta=3$	$\eta=6$
ReT	–	–	3.16×10^3	3.16×10^3	3.16×10^3	3.16×10^3
CC	2.90×10^3	2.91×10^3	5.94×10^3	5.06×10^3	2.28×10^3	2.33×10^3
UC	6.27×10^3	8.09×10^4	5.13×10^3	5.20×10^4	6.64×10^3	7.45×10^4
RF	2.67×10^4	2.16×10^5	2.48×10^4	1.99×10^5	4.52×10^3	3.98×10^4
Total	3.58×10^4	2.99×10^5	4.08×10^4	2.61×10^5	1.66×10^4	1.20×10^5

TABLE II
Complexity in Decoding of the $(64, 49)$ Hermitian Code

Alg.	LCC (BR-RCS)		LCC (ReT-Kötter-RCS)		LCC (ReT-BR-FRF)	
	$\eta=3$	$\eta=6$	$\eta=3$	$\eta=6$	$\eta=3$	$\eta=6$
ReT	–	–	4.25×10^3	4.25×10^3	4.27×10^3	4.22×10^3
CC	2.92×10^3	2.91×10^3	3.60×10^3	2.81×10^3	1.63×10^3	1.66×10^3
UC	4.54×10^3	7.17×10^4	5.02×10^3	5.01×10^4	4.26×10^3	5.36×10^4
RF	3.38×10^4	2.51×10^5	3.16×10^4	2.43×10^5	4.24×10^3	3.75×10^4
Total	4.13×10^4	3.25×10^5	4.62×10^4	3.02×10^5	1.44×10^4	9.69×10^4

We further show the numerical results of decoding complexity and latency. These results were obtained by implementing the decoding schemes using the C programming language and on the AMD R7-5800U CPU. Binary phase-shift keying (BPSK) modulation was used for simulation over the additive white Gaussian noise (AWGN) channel. Tables I and II show the complexity comparison in decoding the $(64, 39)$ and the $(64, 49)$ Hermitian codes, respectively. These results were obtained under a signal-to-noise ratio (SNR) of 7.0 dB. The complexities of ReT, common computation in BR interpolation and common element interpolation in Kötter's interpolation (both marked as CC), uncommon computation in BR interpolation and uncommon element interpolation in Kötter's interpolation (both marked as UC), and root-finding (marked as RF) are listed. The complexity of the proposed LCC decoding with ReT-assisted BR interpolation and FRF (marked as LCC (ReT-BR-FRF)) is compared with the LCC decoding with BR interpolation and RCS algorithm (marked as LCC (BR-RCS)) and the LCC decoding with ReT-assisted Kötter's interpolation and RCS algorithm (marked as LCC (ReT-Kötter-RCS)). Comparing the complexities of common and uncommon computations of the LCC (BR-RCS) and the LCC (ReT-BR-FRF), it can be seen that the latter one yields a lower BR interpolation complexity, indicating that the ReT helps reduce the BR interpolation complexity. This

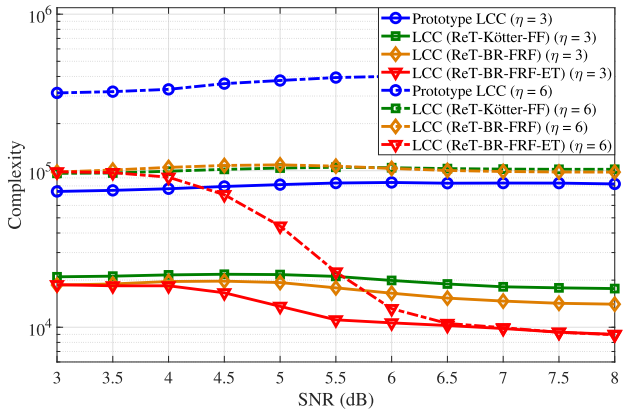

Fig. 3. Decoding complexity in decoding the $(64, 49)$ Hermitian code.

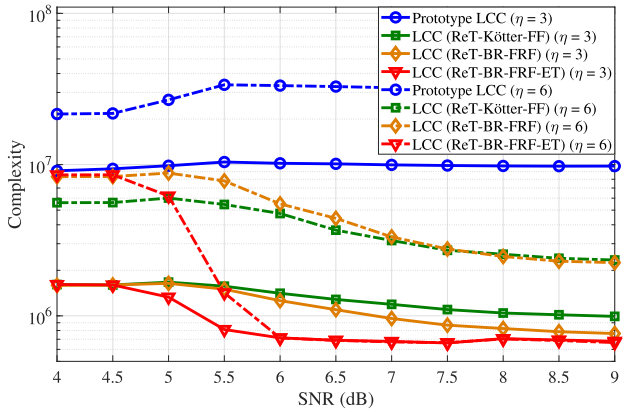

Fig. 4. Decoding complexity in decoding the $(512, 409)$ Hermitian code.

validates the aforementioned complexity analysis of the ReT. Comparing the interpolation complexities of the LCC (ReT-Kötter-RCS) and the LCC (ReT-BR-FRF), it can be seen that although Kötter's interpolation and BR interpolation exhibit a similar theoretical asymptotic complexity, the numerical result of BR interpolation is slightly lower than that of Kötter's interpolation. Comparing their complexities of root-finding, the FRF can provide a significant gain in reducing complexity over the RCS algorithm. The complexity advantage of the proposed LCC (ReT-BR-FRF) emerges for the high-rate codes, This demonstrates its practical merit.

Figs. 3 and 4 show our numerical results of how the decoding complexity varies as the SNR changes in decoding the $(64, 49)$ and the $(512, 409)$ Hermitian codes. The prototype LCC decoding in [20] and the LCC decoding with ReT-assisted Kötter's interpolation and FF (marked as LCC (ReT-Kötter-FF)) in [23] are used as the comparison benchmarks. We have also included simulation results of the proposed LCC decoding with ReT-assisted BR interpolation and FRF, one with early termination through re-encoding and ordering test-vectors (marked as LCC (ReT-BR-FRF-ET)), and another without early termination (i.e., LCC (ReT-BR-FRF)). Recall that π_{thrd} is defined in Section VI. A as the reliability threshold. For the LCC (ReT-BR-FRF-ET), we set $\pi_{\text{thrd}} = 0.6$ and 0.8 for the $(64, 49)$ and the $(512, 409)$

TABLE III

AVERAGE LATENCY (MS) COMPARISON IN LCC DECODING OF THE $(64, 49)$ HERMITIAN CODE

SNR (dB)	Prototype LCC		LCC (ReT-Kötter-FF)		LCC (ReT-BR-FRF)		LCC (ReT-BR-FRF-ET)	
	$\eta = 3$	$\eta = 6$	$\eta = 3$	$\eta = 6$	$\eta = 3$	$\eta = 6$	$\eta = 3$	$\eta = 6$
5.0	45.06	45.33	16.96	16.87	7.00	8.64	19.88	115.11
6.0	44.23	44.71	16.93	17.13	7.29	7.95	7.43	15.16
7.0	44.53	44.91	16.74	16.87	6.81	8.60	4.32	4.88
8.0	44.21	44.39	16.60	16.56	6.57	8.71	2.78	2.85

TABLE IV

AVERAGE LATENCY (S) COMPARISON IN LCC DECODING OF THE $(512, 409)$ HERMITIAN CODE

SNR (dB)	Prototype LCC		LCC (ReT-Kötter-FF)		LCC (ReT-BR-FRF)		LCC (ReT-BR-FRF-ET)	
	$\eta = 3$	$\eta = 6$	$\eta = 3$	$\eta = 6$	$\eta = 3$	$\eta = 6$	$\eta = 3$	$\eta = 6$
5.0	122.98	120.34	32.39	33.72	2.46	2.44	10.15	70.48
6.0	122.93	120.86	32.00	33.47	2.10	2.22	2.29	2.34
7.0	122.76	121.28	32.54	33.52	2.01	2.04	2.08	2.14
8.0	122.77	120.88	33.23	34.04	1.77	1.86	1.68	1.54
9.0	122.79	125.02	33.12	33.57	1.61	1.88	0.97	0.92

Hermitian codes, respectively. As shown in these figures, compared with the prototype LCC decoding, the other three algorithms achieve a much lower complexity. The LCC (ReT-BR-FRF-ET) outperforms the other three algorithms. As the SNR increases, its complexity advantage becomes more significant. When the SNR is high, early termination through re-encoding occurs frequently, maintaining the average decoding complexity at a certain level regardless of whether η is small or large. Note that the LCC (ReT-Kötter-FF) and the LCC (ReT-BR-FRF) have a similar complexity. However, since the uncommon computation in BR interpolation can be performed in parallel, the LCC (ReT-BR-FRF) can yield a much lower latency, as shown in the following decoding latency simulations.

Tables III and IV show our numerical results of decoding latency. It is shown that the other three algorithms yield a lower latency than the prototype LCC decoding, attributed to the contributions of the ReT and the more advanced root-finding algorithms. Furthermore, due to the fully parallel nature of the uncommon computation in BR interpolation, the LCC (ReT-BR-FRF) yields a lower latency than the LCC (ReT-Kötter-FF). When the SNR is low, since the LCC (ReT-BR-FRF-ET) processes the test-vectors in a serial manner and the early termination is rarely achieved, the LCC (ReT-BR-FRF-ET) yields a higher latency over the LCC (ReT-BR-FRF). However, when the SNR is high, the early termination occurs frequently. The LCC (ReT-BR-FRF-ET) yields the lowest latency. As the SNR increases, its latency advantage becomes more significant.

VIII. DECODING PERFORMANCE

This section shows the decoding performance of the proposed algorithms. They are measured as the frame error

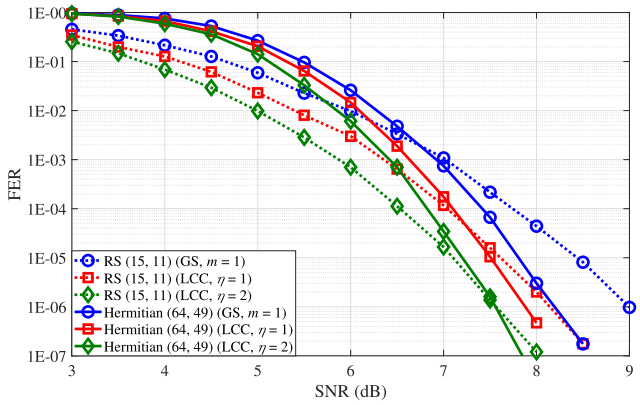

Fig. 5. Decoding performance comparison of the $(15, 11)$ RS code and the $(64, 49)$ Hermitian code.

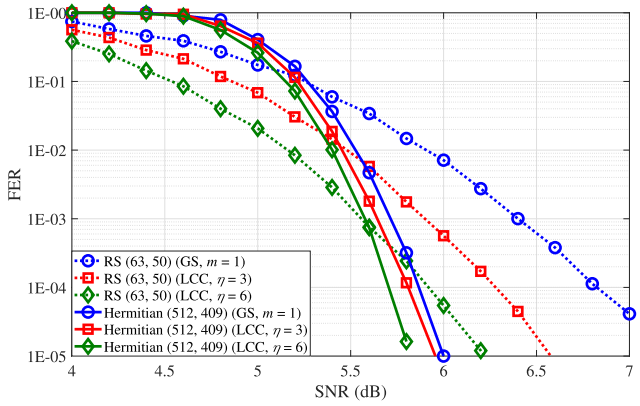

Fig. 6. Decoding performance comparison of the $(63, 50)$ RS code and the $(512, 409)$ Hermitian code.

rate (FER). Note that since the error correction capability of LCC decoding is mainly determined by η, the proposed decoding algorithms maintain the decoding performance of the prototype LCC algorithm. Figs. 5 and 6 compare the $(64, 49)$ and the $(512, 409)$ Hermitian codes with the RS codes defined over the same finite fields. The two RS codes have similar code rates as the Hermitian codes. Over the same finite field, the Hermitian codes are longer with a stronger error-correction capability. Hence, the two Hermitian codes yield better decoding performance over the two RS codes, respectively. Additionally, our results demonstrate that the LCC decoding schemes outperform the GS hard-decision decoding, highlighting the advantages of utilizing soft information to improve error correction capability. As η increases, leading to more test-vectors being decoded, the decoding performance also improves.

IX. Conclusion

This paper has proposed the LCC decoding for Hermitian codes, which is facilitated by both the ReT-assisted BR interpolation and FRF. The ReT has been introduced through defining the Lagrange interpolation polynomials at the Hermitian function fields, reducing both interpolation complexity and latency. The BR interpolation which yields the Gröbner basis of each Chase decoding event is further performed to obtain the interpolation polynomial. The 2^η root-finding processes are further facilitated by the FRF, which can obtain codeword candidates directly from the interpolation outcomes. It eliminates the redundant computation of re-encoding for identifying the most likely codeword candidate. The average LCC decoding complexity can be reduced by assessing the re-encoding codeword and decoding the ordered test-vectors progressively. They both result in early termination when an ML codeword is found. Our simulation results demonstrate that the decoding complexity and latency can be efficiently reduced over existing decoding algorithms for Hermitian codes.

Appendix

Proof of Lemma 1

Since $L_{\Gamma,j}(P_j) = 1$ and $L_{\Gamma,j}(P_{j'}) = 0$, if $j \neq j'$, where $j, j' \in \Gamma$, we have $K_\Gamma(P_j) = \omega_j, \forall j \in \Gamma$. Since $\xi \leq w\lfloor\frac{k-g}{w}\rfloor$ and the affine points defined by Γ form a maximum semi-grid, the ξ points chosen for re-encoding can be categorized into at most $\lfloor\frac{k-g}{w}\rfloor$ groups. Points of each group share the same x-coordinate. For $L_{\Gamma,j}$, we have $|\mathbb{A}(\Gamma)\backslash\{x_j\}| \leq \lfloor\frac{k-g}{w}\rfloor - 1$ and $|\mathbb{B}_{x_j}(\Gamma)\backslash\{y_j\}| = w - 1$. Hence,

$$\begin{aligned}\deg_{1,w_z} L_{\Gamma,j} &\leq w(\lfloor\frac{k-g}{w}\rfloor - 1) + (w+1)(w-1)\\ &\leq k-g-w+w^2-1\\ &= \mu. \end{aligned} \tag{71}$$

Further based on (6), it can be seen that $K_\Gamma \in \mathcal{L}(\mu P_\infty)$.

Proof of Lemma 2

If $\mathcal{Q}_u \in I_{\mathbf{P}^{(u)}}$, $\mathcal{Q}_u$ can be generated by $\mathcal{H}_u^{(\kappa)}$ as in (23). Let $\mathcal{H}_u'^{(\kappa)}(x, y, z) = \mathcal{H}_u^{(\kappa)}(x, y, z + K_\Gamma)$ as

$$\mathcal{H}_u'^{(\kappa)} = G^{1-\kappa}(z + K_\Gamma - K_u)^\kappa. \tag{72}$$

Since $K_u(P_j) - K_\Gamma(P_j) = \omega_j^{(u)} - h_j = z_j$, $\mathcal{H}_u'^{(\kappa)}$ interpolates the transformed interpolation points of $\mathbf{P}'_u$ with a multiplicity of one, i.e. $\mathcal{H}_u'^{(\kappa)} \in I_{\mathbf{P}'_u}$. Therefore, if $\mathcal{Q}_u \in I_{\mathbf{P}^{(u)}}$, $\mathcal{Q}'_u \in I_{\mathbf{P}'_u}$, and vice versa.

Since $\deg_{1,w_z} K_\Gamma \leq \mu$, we have $\deg_{1,w_z} z = \deg_{1,w_z}(z + K_\Gamma)$ and $\deg_{1,w_z} \mathcal{Q}'_u = \deg_{1,w_z} \mathcal{Q}_u$. Let us assume that $\mathcal{Q}_u$ is the minimum polynomial of $I_{\mathbf{P}^{(u)}}$. If there exists $N' \in I_{\mathbf{P}'_u}$ and exhibits $\deg_{1,w_z} N' < \deg_{1,w_z} \mathcal{Q}'_u$, $N = N'(x, y, z - K_\Gamma) \in I_{\mathbf{P}^{(u)}}$ and $\deg_{1,w_z} N = \deg_{1,w_z} N'$. It leads to $\deg_{1,w_z} N < \deg_{1,w_z} \mathcal{Q}_u$, which contradicts to $\mathcal{Q}_u$ being the minimum polynomial of $I_{\mathbf{P}^{(u)}}$. Therefore, $\mathcal{Q}'_u$ is the minimum polynomial of $I_{\mathbf{P}'_u}$, and vice versa.

Proof of Lemma 3

Since G_{Γ^c} and $z - \mathcal{K}_{\Gamma^c}^{(u)}$ interpolate all points of $\widetilde{\mathbf{P}}_u$, $\widetilde{M}_u^{(s)} \in I_{\widetilde{\mathbf{P}}_u}$. For any $Q \in I_{\widetilde{\mathbf{P}}_u}$, there exists $\mathcal{V}_0, \mathcal{V}_1 \in R$ such that $Q = \mathcal{V}_0 \widetilde{M}_u^{(0)} + \mathcal{V}_1 \widetilde{M}_u^{(w)}$. Since $\mathcal{V}_\kappa = \sum_{\iota=0}^{w-1} \mathcal{W}_{\kappa,\iota} y^\iota$, where $\mathcal{W}_{\kappa,\iota} \in \mathbb{F}_q[x]$ and $\kappa = 0$ or 1, $Q = \sum_{\kappa=0}^{1} \sum_{\iota=0}^{w-1} \mathcal{W}_{\kappa,\iota} y^\iota \widetilde{\mathcal{H}}_u^{(\kappa)}$. Therefore, $I_{\widetilde{\mathbf{P}}_u}$ can be generated by $\mathcal{M}_{\widetilde{\mathbf{P}}_u}$.

PROOF OF LEMMA 4

Based on (27), $\deg_{1,\underline{w}_z} G_\Gamma = \xi$. Based on (32), it is known that $\mathcal{Q}_u = G_\Gamma \widetilde{\mathcal{Q}}_u(x, y, \frac{z-K_\Gamma}{G_\Gamma})$. Let $\widetilde{\mathcal{Q}}_u(x,y,z) = \widetilde{\mathcal{Q}}_u^{(0)}(x,y) + \widetilde{\mathcal{Q}}_u^{(1)}(x,y)z$. Since $\deg_{1,w_z} K_\Gamma \le \mu$,

$$\begin{aligned}\deg_{1,w_z} \mathcal{Q}_u &= \xi + \deg_{1,w_z} \widetilde{\mathcal{Q}}_u(x, y, \frac{z-K_\Gamma}{G_\Gamma}) \\ &= \xi + \deg_{1,w_z}(\widetilde{\mathcal{Q}}_u^{(0)} + \widetilde{\mathcal{Q}}_u^{(1)} \frac{z-K_\Gamma}{G_\Gamma}) \\ &= \xi + \max_{0\le j\le 1}\{\deg_{1,w_z}[\widetilde{\mathcal{Q}}_u^{(j)} + j(w_z - \xi)]\}. \quad (73)\end{aligned}$$

Therefore, $\deg_{1,w_z} \mathcal{Q}_u = \deg_{1,\widetilde{w}_z} G_\Gamma + \deg_{1,\widetilde{w}_z} \widetilde{\mathcal{Q}}_u$.

PROOF OF LEMMA 5

The mapping of (32) shows that if $\widetilde{\mathcal{Q}}_u \in I_{\widetilde{\mathbf{P}}_u}$, $\mathcal{Q}'_u \in I_{\mathbf{P}'_u}$. Let us assume that $\widetilde{\mathcal{Q}}_u$ is the minimum polynomial of $I_{\widetilde{\mathbf{P}}_u}$. If there exists a polynomial $N' \in I_{\mathbf{P}'_u}$ that satisfies $\deg_{1,w_z} N' < \deg_{1,w_z} \mathcal{Q}'_u$, $G_\Gamma(x)\widetilde{N}(x,y,\frac{z}{G_\Gamma}) = N'(x,y,z)$. Based on Lemma 4, $\deg_{1,\widetilde{w}_z} G_\Gamma + \deg_{1,\widetilde{w}_z} \widetilde{N} < \deg_{1,\widetilde{w}_z} G_\Gamma + \deg_{1,\widetilde{w}_z} \widetilde{\mathcal{Q}}_u$, which contradicts to $\widetilde{\mathcal{Q}}_u$ being the minimum polynomial of $I_{\widetilde{\mathbf{P}}_u}$. Hence, if $\widetilde{\mathcal{Q}}_u$ is the minimum polynomial of $I_{\widetilde{\mathbf{P}}_u}$, $\mathcal{Q}'_u(x,y,z) = G_\Gamma(x)\widetilde{\mathcal{Q}}_u(x,y,\frac{z}{G_\Gamma})$ is the minimum polynomial of $I_{\mathbf{P}'_u}$. Further based on Lemma 2, if $\mathcal{Q}'_u$ is the minimum polynomial of $I_{\mathbf{P}'_u}$, $\mathcal{Q}_u(x,y,z) = \mathcal{Q}'_u(x,y,z-K_\Gamma)$ is also the one of $I_{\mathbf{P}^{(u)}}$.

PROOF OF LEMMA 6

Since $L_{\mathcal{D}_u,j}(P_j) = 1$ and $L_{\mathcal{D}_u,j}(P_{j'}) = 0$, if $j \ne j'$, where $j, j' \in \mathcal{D}_u$, we have $\mathcal{K}_{\mathcal{D}_u}(P_j) = \widetilde{c}_j^{(u)}, \forall j \in \mathcal{D}_u$.

The symbols of $\{\widetilde{c}_j^{(u)} \mid \forall j \in \mathcal{D}_u\}$ can be understood as erasing $n - |\mathcal{D}_u|$ symbols of $\underline{\widetilde{c}}_u$, or the codeword symbols of a punctured Hermitian code. For a punctured Hermitian code, where the affine points associated with the unpunctured positions form a maximum semi-grid, the corresponding message polynomial can be computed using Lagrange interpolation [33]. Since the affine points defined by $\mathcal{D}_u$ form a maximum semi-grid, $\mathcal{K}_{\mathcal{D}_u} \in \mathcal{L}(\mu P_\infty)$. Let $\underline{\varsigma} = (\varsigma_0, \varsigma_1, \ldots, \varsigma_{n-1})$ where $\varsigma_j = \mathcal{K}_{\mathcal{D}_u}(P_j), \forall j \in [0:n-1]$. Since the designed distance of Hermitian codes is $d^* = n-k-g+1$, if $\underline{\varsigma} \ne \underline{\widetilde{c}}_u$, we have

$|\{j \mid \widetilde{c}_j^{(u)} \ne \varsigma_j\}| \ge d^*$. If $|\mathcal{D}_u| \ge w\lceil\frac{k+g}{w}\rceil \ge k+g$, $|\{j \mid \widetilde{c}_j^{(u)} \ne \varsigma_j\}| \le n-k-g$. This is a contradiction. Therefore, $\underline{\varsigma} = \underline{\widetilde{c}}_u$. As a result, the $n - |\mathcal{D}_u|$ erasures of $\underline{\widetilde{c}}_u$ including the symbols associated with Λ_u can be recovered as in (57) and (58).

REFERENCES

[1] V. D. Goppa, "Codes associated with divisors," *Problemy Peredachi Informatsii*, vol. 13, no. 1, pp. 33–39, 1977.

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

[3] S. Sakata, J. Justesen, Y. Madelung, H. E. Jensen, and T. Hoholdt, "Fast decoding of algebraic-geometric codes up to the designed minimum distance," *IEEE Trans. Inf. Theory*, vol. 41, no. 6, pp. 1672–1677, Nov. 1995.

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

[5] R. M. 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.

[6] X.-W. Wu and P. H. Siegel, "Efficient root-finding algorithm with application to list decoding of algebraic-geometric codes," *IEEE Trans. Inf. Theory*, vol. 47, no. 6, pp. 2579–2587, Sep. 2001.

[7] L. Chen, R. A. Carrasco, M. Johnston, and E. G. Chester, "Efficient factorisation algorithm for list decoding algebraic-geometric and Reed–Solomon codes," in *Proc. IEEE Int. Conf. Commun.*, Glasgow, U.K., Jun. 2007, pp. 851–856.

[8] R. Kötter, "On algebraic decoding of algebraic-geometric and cyclic codes," Ph.D. thesis, Dept. Elect. Eng., Univ. Linköping, Linköping, Sweden, 1996.

[9] 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, Sep. 2008.

[10] K. Lee and M. E. O'Sullivan, "List decoding of Hermitian codes using Gröbner bases," *J. Symbolic Comput.*, vol. 44, no. 12, pp. 1662–1675, Dec. 2009.

[11] J. S. R. Nielsen and P. Beelen, "Sub-quadratic decoding of one-point Hermitian codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 6, pp. 3225–3240, Jun. 2015.

[12] P. Beelen, J. Rosenkilde, and G. Solomatov, "Fast decoding of AG codes," *IEEE Trans. Inf. Theory*, vol. 68, no. 11, pp. 7215–7232, Nov. 2022.

[13] P. Beelen and V. Neiger, "Faster list decoding of AG codes," 2023, *arXiv:2304.07083*.

[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] Y. Wan, L. Chen, and F. Zhang, "Guruswami–Sudan decoding of elliptic codes through module basis reduction," *IEEE Trans. Inf. Theory*, vol. 67, no. 11, pp. 7197–7209, Nov. 2021.

[16] 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.

[17] L. Chen, R. Carrasco, and M. Johnston, "Soft-decision list decoding of Hermitian codes," *IEEE Trans. Commun.*, vol. 57, no. 8, pp. 2169–2176, Aug. 2009.

[18] K. Lee and M. E. O'Sullivan, "Algebraic soft-decision decoding of Hermitian codes," *IEEE Trans. Inf. Theory*, vol. 56, no. 6, pp. 2587–2600, Jun. 2010.

[19] 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.

[20] S. Wu, L. Chen, and M. Johnston, "Interpolation-based low-complexity chase decoding algorithms for Hermitian codes," *IEEE Trans. Commun.*, vol. 66, no. 4, pp. 1376–1385, Apr. 2018.

[21] L. Chen and F. Zhang, "Algebraic chase decoding of elliptic codes through computing the Gröbner basis," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Espoo, Finland, Jun. 2022, pp. 180–185.

[22] J. Zhu and X. Zhang, "Factorization-free low-complexity chase soft-decision decoding of Reed–Solomon codes," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Taipei, Taiwan, May 2009, pp. 2677–2680.

[23] J. Liang and L. Chen, "Low-complexity chase decoding of Hermitian codes with re-encoding transform and fast factorization," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Kuala Lumpur, Malaysia, Dec. 2023, pp. 1680–1685.

[24] 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.

[25] X. Ma and S. Tang, "Correction to 'an efficient maximum-likelihood-decoding algorithm for linear block codes with algebraic decoder,'" *IEEE Trans. Inf. Theory*, vol. 58, no. 6, p. 4073, Jun. 2012.

[26] J. Zhao, L. Chen, X. Ma, and M. Johnston, "Progressive algebraic chase decoding algorithms for Reed–Solomon codes," *IET Commun.*, vol. 10, no. 12, pp. 1416–1427, Aug. 2016.

[27] J. Xing, L. Chen, and M. Bossert, "Low-complexity chase decoding of Reed–Solomon codes using module," *IEEE Trans. Commun.*, vol. 68, no. 10, pp. 6012–6022, Oct. 2020.

[28] I. F. Blake, C. Heegard, T. Høholdt, and V. K. Wei, "Algebraic-geometry codes," *IEEE Trans. Inf. Theory*, vol. 44, no. 6, pp. 2596–2618, Jan. 1998.
[29] R. Nielsen, "List decoding of linear block codes," Ph.D. thesis, Dept. Math., Tech. Univ. Denmark, Kongens Lyngby, Denmark, 2001.
[30] H. Stichtenoth, *Algebraic Function Fields and Codes*, 2nd ed., New York, NY, USA: Springer-Verlag, 2009.
[31] T. Høholdt and R. Nielsen, "Decoding Hermitian codes with Sudan's algorithm," in *Applied Algebra, Algebraic Algorithms and Error Correcting Codes* (Lecture Notes in Computer Science), vol. 1719, M. Fossorier, S. Lin, and A. Pole, Eds., Berlin, Germany: Springer-Verlag, 1999, pp. 260–269.
[32] T. Mulders and A. Storjohann, "On lattice reduction for polynomial matrices," *J. Symbolic Comput.*, vol. 35, no. 4, pp. 377–401, Apr. 2003.
[33] P. Beelen, J. Rosenkilde, and G. Solomatov, "Fast encoding of AG codes over cab curves," *IEEE Trans. Inf. Theory*, vol. 67, no. 3, pp. 1641–1655, Mar. 2021.

Jiwei Liang received the B.E. degree in electronic engineering from Southeast University, Nanjing, China, in 2013, and the M.S. degree in communication engineering from Guilin University of Electronic Technology, Guilin, China, in 2017. He is currently pursuing the Ph.D. degree in information and communication engineering with Sun Yat-sen University, Guangzhou, China.

In 2013, he was with Avonaco Communication Systems Company Ltd., Suzhou, China, as a DSP Engineer, where he was involved in the development of multimedia communication systems. From 2017 to 2021, he was with Allwinner Technology Company Ltd., Zhuhai, China, as a Firmware Engineer, where he was involved in the development of WLAN/Bluetooth combo chip. His research interests include channel coding and data communication.

Jianguo Zhao received the B.Sc. degree in information engineering and the M.Sc. degree in information and communication engineering from Sun Yat-sen University, Guangzhou, China, in 2021 and 2024, respectively. His research interests include channel coding and data communications.

Li Chen (Senior Member, IEEE) 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 returned China, as a Lecturer with the School of Information Science and Technology, Sun Yat-sen University, Guangzhou. From 2011 to 2012, he was a Visiting Researcher with the Institute of Network Coding, The Chinese University of Hong Kong, where he was an Associate Professor and a Professor from 2011 and 2016. Since 2013, he has been the Associate Head of the Department of Electronic and Communication Engineering (ECE). From July 2015 to October 2015, he was a Visitor with the Institute of Communications Engineering, Ulm University, Germany. From October 2015 to June 2016, he was a Visiting Associate Professor with the Department of Electrical Engineering, University of Notre Dame, USA. From 2017 to 2020, he was the Deputy Dean of the School of Electronics and Communication Engineering. His research interests include information theory, error-correction codes, and data communications. He is a Senior Member of Chinese Institute of Electronics (CIE), a member of the IEEE Information Theory Society Board of Governors and its External Nomination Committee, and the Chair of its Conference Committee and the IEEE Information Theory Society Guangzhou Chapter. He has been organizing several international conferences and workshops, including the 2018 IEEE Information Theory Workshop (ITW) at Guangzhou and the 2022 IEEE East Asian School of Information Theory (EASIT) at Shenzhen, for which he is the General Co-Chair. He is also the TPC Co-Chair of the 2022 IEEE/CIC International Conference on Communications in China (ICCC) at Foshan. He was an Associate Editor of IEEE TRANSACTIONS ON COMMUNICATIONS and is currently an Associate Editor of IEEE TRANSACTIONS ON INFORMATION THEORY.