Efficient Root-Finding for Interpolation-Based Decoding of Elliptic and Hyperelliptic Codes

Jianguo Zhao[†], Jiwei Liang[†] and Li Chen[‡§]
[†] School of System Science and Engineering, Sun Yat-sen University, Guangzhou, China
[‡] School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou, China
[§] Guangdong Province Key Laboratory of Information Security Technology, Guangzhou, China
Email: zhaojg5@mail2.sysu.edu.cn, liangjw59@mail2.sysu.edu.cn, chenli55@mail.sysu.edu.cn

2024 IEEE International Symposium on Information Theory (ISIT) | 979-8-3503-8284-6/24/$31.00 ©2024 IEEE | DOI: 10.1109/ISIT57864.2024.10619084

***Abstract*—This paper proposes an efficient root-finding algorithm for the interpolation-based unique decoding of one-point elliptic and hyperelliptic codes. Instead of finding a message polynomial, it directly computes a codeword from the interpolation polynomial. It first determines the error positions through the error locator polynomial that is contained in the interpolation polynomial. Subsequently, the corresponding codeword symbols are determined based on the root-finding equation that is reformulated as a linear system of univariate polynomials. The proposed algorithm demonstrates superiority to the Roth-Ruckenstein (RR) algorithm, especially in the scenarios that the decoder is required to output a codeword and the re-encoding transform (ReT) technique is employed.**

***Index Terms*—Algebraic-geometric codes, elliptic curves, hyperelliptic curves, interpolation-based decoding, root-finding**

I. Introduction

Constructed over an algebraic curve, algebraic-geometric (AG) codes [1] have a good minimum distance property and a codeword length that can exceed the size of the underlying finite field. Elliptic and hyperelliptic codes are a class of AG codes. They are instances of codes from the Miura-Kamiya curves [2], which exhibit simpler encoding and decoding computations than most other AG codes.

Decoding algorithms for AG codes can be categorized into syndrome-based [3]–[5] and interpolation-based [6] [7] approaches. The former first determines an error-locator function and then obtains an estimated codeword through computing the error values over the error positions [8]–[10]. The later first constructs an interpolation polynomial and then obtains a list of estimated message functions, i.e., achieves list decoding through finding the roots of the polynomial. When both the interpolation multiplicity and list size are one, the interpolation-based decoding turns into a unique decoding that can correct up to $\lfloor (d^* - g - 1)/2 \rfloor$ errors[1] [12], where d^* and g denote the designed distance and genus of the code, respectively. This setting frequently occurs in the algebraic Chase decoding (ACD) [13]–[15], where the complexity of root-finding is prominent as it needs to be executed for a set of interpolation polynomials. To realize the root-finding, the Roth-Ruckenstein (RR) algorithm [16] and its divide-and-conquer variants [17]–[19] can be utilized. They output message functions. However, there are certain scenarios that require the decoder to output codewords, such as when the encoder employs a systematic encoding scheme or the likelihood of estimated codewords needs to be measured (e.g., in the ACD). In these scenarios, the RR algorithm has to encode the output messages in order to obtain the corresponding codewords. Moreover, when the re-encoding transform (ReT) [20]–[22] is employed to facilitate the interpolation, the RR algorithm requires a restoration of the interpolation polynomial, which involves multiplying two polynomials with a large degree.

This paper proposes an efficient root-finding algorithm for the interpolation-based unique decoding of one-point elliptic and hyperelliptic codes. Instead of finding a message polynomial, it directly computes a codeword from the interpolation polynomial. This makes the output feature of interpolation-based decoding consistent with that of syndrome-based decoding. It begins with determining the error positions through the error locator polynomial that is contained in the interpolation polynomial. Subsequently, the corresponding codeword symbols are determined based on the root-finding equation that is reformulated as a linear system of univariate polynomials. Compared to the RR algorithm, the proposed algorithm eliminates the computations for encoding the message polynomial to obtain a codeword and restoring the interpolation polynomial in the context of ReT. Furthermore, it also enables computational parallelism which would be welcomed by hardware implementations.

Notations

Let $\mathbb{F}$ denote a Galois field (GF). The univariate, bivariate and trivariate polynomial rings over $\mathbb{F}$ are denoted as $\mathbb{F}[x]$, $\mathbb{F}[x,y]$ and $\mathbb{F}[x,y,z]$, respectively. Given a polynomial[2]

$$h(x,y,z) = \sum_{i_x \geqslant 0} \sum_{i_y \geqslant 0} \sum_{i_z \geqslant 0} h_{i_x,i_y,i_z} x^{i_x} y^{i_y} z^{i_z},$$

its (w_x, w_y, w_z)-weighted degree is defined as

$$\deg_{w_x,w_y,w_z} h = \max_{i_x,i_y,i_z} \{w_x i_x + w_y i_y + w_z i_z : h_{i_x,i_y,i_z} \neq 0\}.$$

Its (w_x, w_y)-weighted degree, y-degree and z-degree are further defined as $\deg_{w_x,w_y} h = \deg_{w_x,w_y,0} h$, $\deg_y h = \deg_{0,1,0} h$ and $\deg_z h = \deg_{0,0,1} h$, respectively.

[1] In most cases, it corrects up to $\lfloor (d^* - 1)/2 \rfloor$ errors [11].

[2] A univariate or bivariate polynomial can be viewed as a special instance of trivariate polynomials.

II. Preliminaries

A. Elliptic and Hyperelliptic Codes

Given an integer $g \geqslant 1$, let $H(X,Y)$ be an absolutely irreducible polynomial in the form of

$$H(X,Y) = Y^2 + H_1(X)Y + H_0(X), \tag{1}$$

where $H_0(X), H_1(X) \in \mathbb{F}[X]$, $\deg H_1 \leqslant g$ and $\deg H_0 = b := 2g+1$. An affine elliptic or hyperelliptic curve $\mathcal{X}$ over $\mathbb{F}$ is defined by $H(X,Y) = 0$. When $g = 1$, it becomes an elliptic curve. It can be shown that the genus of $\mathcal{X}$ is g. The set of rational affine points on $\mathcal{X}$ is denoted as $\mathcal{P} = \{P_1, P_2, ..., P_n\}$, where $P_j = (\alpha_j, \beta_j)$ satisfies $H(P_j) = 0$ for all $j = 1, 2, ..., n$. The unique rational point at infinity is denoted as P_∞. According to the Serre bound [23, Chapter 5], $n \leqslant |\mathbb{F}| + g\lfloor 2\sqrt{|\mathbb{F}|} \rfloor$. For each point $P_j \in \mathcal{P}$, there exists a unique point $(\alpha_j, -\beta_j - H_1(\alpha_j)) \in \mathcal{P}$ that shares the same X-coordinate with P_j. Let $\mathsf{s}[j]$ denote its index. For the sake of simplicity, this paper assumes that $P_j \neq P_{\mathsf{s}[j]}$.[3]

Let x and y denote two functions in the function field of $\mathcal{X}$ which satisfy $H(x,y) = 0$. Based on (1), any nonzero polynomial $h(x,y) \in \mathbb{F}[x,y]$ has poles only at P_∞ with a pole order of $\deg_{2,b} h$ [24]. Moreover, if $\deg_y h \geqslant 2$, there exists another polynomial $h'(x,y) \in \mathbb{F}[x,y]$ with $\deg_y h' < 2$ such that $h(x,y) = h'(x,y)$. Consequently, $\mathbb{F}[x,y]$ is equivalent to the following polynomial set

$$\begin{aligned}\mathbb{F}[x][y]_2 &:= \{h \in \mathbb{F}[x,y] : \deg_y h < 2\} \\ &= \{h_0(x) + h_1(x)y : h_0, h_1 \in \mathbb{F}[x]\}.\end{aligned}$$

Based on the above prerequisites, the Riemann-Roch space [23, Chapter 1] of divisor uP_∞ can be represented as

$$\mathcal{L}(uP_\infty) = \{f \in \mathbb{F}[x][y]_2 : \deg_{2,b} f \leqslant u\}. \tag{2}$$

A one-point evaluation elliptic or hyperelliptic code of length n is defined as

$$\mathbb{C}(\mathcal{P}, u) = \{(f(P_1), f(P_2), ..., f(P_n)) : f(x,y) \in \mathcal{L}(uP_\infty)\}.$$

When $2g - 2 < u < n$, $\mathbb{C}(\mathcal{P}, u)$ has a dimension of $k = u - g + 1$ and a designed distance of $d^* = n - u$.

B. Interpolation-Based Unique Decoding

Considering both the interpolation multiplicity and the list size are one, the interpolation-based decoding can be formulated as follows. Given a received word $\mathbf{r} = (r_1, r_2, ..., r_n) \in \mathbb{F}^n$ of $\mathbb{C}(\mathcal{P}, u)$, let $\mathbf{P} = \{(\alpha_j, \beta_j, r_j) : j = 1, 2, ..., n\}$. The interpolation computes a nonzero polynomial $Q(x,y,z) = Q_0(x,y) + Q_1(x,y)z$ that interpolates points of $\mathbf{P}$, i.e.,

$$Q(\alpha_j, \beta_j, r_j) = 0, \text{ for all } j = 1, 2, ..., n. \tag{3}$$

Meanwhile, $\deg_{2,b,u} Q$ is minimal among all polynomials in

$$\mathbb{F}[x,y][z]_2 := \{h \in \mathbb{F}[x,y,z] : \deg_z h < 2\}$$

that interpolate points of $\mathbf{P}$. The root-finding further computes the z-root $\hat{f}(x,y)$ of $Q(x,y,z)$ such that

$$Q_0(x,y) + Q_1(x,y)\hat{f}(x,y) = 0. \tag{4}$$

Note that the decoder can produce a valid output if and only if the root-finding equation (4) has a solution in $\mathcal{L}(uP_\infty)$. Based on the equivalence of $\mathbb{F}[x,y]$ and $\mathbb{F}[x][y]_2$, the aforementioned bivariate polynomials can be represented as

$$\begin{aligned}\hat{f}(x,y) &= \hat{f}_0(x) + \hat{f}_1(x)y, \\ Q_i(x,y) &= Q_{0,i}(x) + Q_{1,i}(x)y, \text{ for } i = 1, 2.\end{aligned} \tag{5}$$

III. The Error Locator Polynomial

This section first shows that $Q_1(x,y)$ is an error locator polynomial. The error positions can be determined through finding the zeros of $Q_1(x,y)$. Subsequently, the root-finding equation (4) is reformulated as a linear system over $\mathbb{F}[x]$. The determinant of the coefficient matrix can be viewed as another form of the error locator polynomial. Properties of this polynomial are characterized, providing the foundation for determining the codeword symbols over the error positions.

A. Determination of Error Positions

Assume that there exists $\hat{f}(x,y) \in \mathcal{L}(uP_\infty)$ such that (4) holds. In conjunction with (3), we have

$$\begin{cases}\hat{f}(P_j) = r_j, & \text{if } Q_1(P_j) \neq 0; \\ Q_0(P_j) = 0, & \text{if } Q_1(P_j) = 0.\end{cases} \tag{6}$$

This implies that r_j is a codeword symbol if $Q_1(P_j) \neq 0$; otherwise, r_j may be erroneous. Hence, $Q_1(x,y)$ is referred to as an error locator polynomial [25]. Its zeros indicate all the potential error positions.

Let us define the error position set as

$$\mathcal{E} = \{j \in \{1, 2, ..., n\} : Q_1(P_j) = 0\}. \tag{7}$$

Its complement set is denoted as $\bar{\mathcal{E}} := \{1, 2, ..., n\} \backslash \mathcal{E}$. Let $\mathcal{M}_\mathcal{E}$ denote the set of polynomials in $\mathbb{F}[x][y]_2$ that interpolate P_j for all $j \in \mathcal{E}$, i.e.,

$$\mathcal{M}_\mathcal{E} = \{h \in \mathbb{F}[x][y]_2 : h(P_j) = 0 \text{ for all } j \in \mathcal{E}\}.$$

It is demonstrated in *Lemma 1* that the error locator polynomial $Q_1(x,y)$ is a minimal polynomial in $\mathcal{M}_\mathcal{E}$.

Lemma 1. *If there exists $\hat{f}(x,y) \in \mathbb{F}[x][y]_2$ such that (4) holds, then $Q_1(x,y)$ is a nonzero polynomial of the minimal $(2,b)$-weighted degree among all polynomials in $\mathcal{M}_\mathcal{E}$, denoted as $Q_1(x,y) = \min \mathcal{M}_\mathcal{E}$.*

Due to space limitations, the proof of *Lemma 1* is omitted. It can be proven by contradiction with the assumption that there exists a nonzero polynomial $Q_1'(x,y) \in \mathcal{M}_\mathcal{E}$ such that $\deg_{2,b} Q_1' < \deg_{2,b} Q_1$. The following *Lemma 2* characterizes the $(2,b)$-weighted degree of the minimal polynomial in $\mathcal{M}_\mathcal{E}$.

Lemma 2. *Let $h(x,y) = \min \mathcal{M}_\mathcal{E}$. Then $|\mathcal{E}| \leqslant \deg_{2,b} h \leqslant |\mathcal{E}| + g$.*

[3]Despite we overlook this case, the proposed algorithm can accommodate it with a slight modification.

The lower bound in *Lemma 2* is obvious and the upper bound can be derived from Riemann's theorem [23, Chapter 1]. Based on *Lemmas 1* and *2*, the degree relation

$$\deg_{2,b} Q_1 - |\mathcal{E}| \leqslant g \tag{8}$$

is a necessary condition for $\hat{f}(x,y) \in \mathbb{F}[x][y]_2$. Moreover, for $\hat{f}(x,y) \in \mathcal{L}(uP_\infty)$, it is required that

$$\deg_{2,b} Q_0 - \deg_{2,b} Q_1 \leqslant u. \tag{9}$$

Therefore, if either of conditions (8) and (9) is not satisfied, the decoder declares *failure*. In fact, the numerical results in Tab. II show that the decoder almost always produces a valid output when these two conditions are both satisfied.

The following remark characterizes the computational complexity of determining $\mathcal{E}$. Note that in this paper, complexity is measured as the number of finite field multiplications.

Remark 3. *Determining the error position set $\mathcal{E}$ as in* (7) *involves evaluating $Q_1(x,y)$ at P_j for all $j = 1,2,...,n$. Since $\alpha_j = \alpha_{\mathsf{s}[j]}$, both $Q_1(P_j)$ and $Q_1(P_{\mathsf{s}[j]})$ can be obtained based on $Q_{0,1}(\alpha_j)$ and $Q_{1,1}(\alpha_j)$, i.e., $Q_1(P_{j'}) = Q_{0,1}(\alpha_j) + Q_{1,1}(\alpha_j)\beta_{j'}$ for $j' = j, \mathsf{s}[j]$. If* (8) *holds, computing $n/2$ pairs of $Q_{0,1}(\alpha_j)$ and $Q_{1,1}(\alpha_j)$ can be realized through less than $n(|\mathcal{E}| + g)/2$ multiplications. Note that $Q_{0,1}(\alpha_j)$ and $Q_{1,1}(\alpha_j)$ may be reused in the subsequent computations.*

B. Root-Finding Equations over $\mathbb{F}[x]$

With (5), the root-finding equation (4) can be rewritten as

$$\begin{aligned} &Q_{0,0}(x) + Q_{0,1}(x)\hat{f}_0(x) + Q_{1,1}(x)\hat{f}_1(x)y^2 \\ =& -(Q_{1,0}(x) + Q_{1,1}(x)\hat{f}_0(x) + Q_{0,1}(x)\hat{f}_1(x))y. \end{aligned}$$

Based on $H(x,y) = 0$ and (1), it can be transformed into

$$\begin{aligned} &Q_{0,0}(x) + Q_{0,1}(x)\hat{f}_0(x) + V_0(x)\hat{f}_1(x) \\ =& -(Q_{1,0}(x) + Q_{1,1}(x)\hat{f}_0(x) + V_1(x)\hat{f}_1(x))y, \end{aligned} \tag{10}$$

where

$$\begin{aligned} V_0(x) &= -H_0(x)Q_{1,1}(x), \\ V_1(x) &= Q_{0,1}(x) - H_1(x)Q_{1,1}(x). \end{aligned} \tag{11}$$

The above equation (10) can be viewed as a linear system over the univariate polynomial ring $\mathbb{F}[x]$:

$$\begin{bmatrix} Q_{0,1}(x) & V_0(x) \\ Q_{1,1}(x) & V_1(x) \end{bmatrix} \begin{bmatrix} \hat{f}_0(x) \\ \hat{f}_1(x) \end{bmatrix} = -\begin{bmatrix} Q_{0,0}(x) \\ Q_{1,0}(x) \end{bmatrix}. \tag{12}$$

Therefore, finding the estimated message polynomial $\hat{f}(x,y)$ has been transformed into finding the solution $(\hat{f}_0(x), \hat{f}_1(x))$ to the $\mathbb{F}[x]$-linear equations. Let $D(x)$ denote the determinant of the 2×2 matrix in (12), i.e.,

$$D(x) = V_1(x)Q_{0,1}(x) - V_0(x)Q_{1,1}(x). \tag{13}$$

Equation (12) has a unique solution only if $D(x) \neq 0$. It is demonstrated by *Theorem 5* that $D(x) \neq 0$. In fact, $D(x)$ can be understood as another form of the error locator polynomial. To prove *Theorem 5*, the following *Lemma 4* should be introduced first.

Lemma 4. *Let $\mathsf{S}[\mathcal{E}] = \{j \in \mathcal{E} : \mathsf{s}[j] \in \mathcal{E}\}$. Then we have:*

i) *For any $j \in \mathsf{S}[\mathcal{E}]$, $(x - \alpha_j) \mid Q_{i_y,i_z}(x)$ for $i_y = 0,1$ and $i_z = 0,1$;*
ii) *For any $j \in \mathcal{E}\backslash\mathsf{S}[\mathcal{E}]$, $Q_{1,1}(\alpha_j) \neq 0$.*

Theorem 5. *The determinant polynomial $D(x)$ defined in* (13) *satisfies the following three properties:*

i) *$\deg D = \deg_{2,b} Q_1$;*
ii) *$D(x)$ is divisible by $\prod_{j\in\mathcal{E}}(x - \alpha_j)$;*
iii) *When $Q_1(x,y) = \min \mathcal{M}_\mathcal{E}$, $D(\alpha) \neq 0$ for any $\alpha \in \mathbb{F}$ that satisfies $\alpha \neq \alpha_j$ for all $j \in \mathcal{E}$.*

Proof: i) It can be observed that

$$\deg_{2,b} Q_1 = \max\{2\deg Q_{0,1}, 2\deg Q_{1,1} + b\}.$$

Then we have $2\deg Q_{0,1} \neq 2\deg Q_{1,1} + b$ since $b = 2g + 1$ is an odd number. Note that $\deg H_0 = b$ and $\deg H_1 \leqslant g$. If $2\deg Q_{0,1} > 2\deg Q_{1,1} + b$, then $\deg H_1 + \deg Q_{1,1} + \deg Q_{0,1} < 2\deg Q_{0,1}$ and thus $\deg D = 2\deg Q_{0,1}$. Otherwise, $\deg H_1 + \deg Q_{1,1} + \deg Q_{0,1} < 2\deg Q_{1,1} + b$ and thus $\deg D = 2\deg Q_{1,1} + b$. Therefore, $\deg D = \deg_{2,b} Q_1$.

The proofs of properties ii) and iii) involve the following equations (14) and (15). Given $\alpha \in \mathbb{F}$,

$$D(\alpha) = Q_{0,1}(\alpha)^2 - H_1(\alpha)Q_{1,1}(\alpha)Q_{0,1}(\alpha) + H_0(\alpha)Q_{1,1}(\alpha)^2 \tag{14}$$

If $Q_{1,1}(\alpha) \neq 0$, let $\Delta_\alpha = -Q_{0,1}(\alpha)/Q_{1,1}(\alpha)$. Then (14) can be written as

$$\begin{aligned} D(\alpha) &= Q_{1,1}(\alpha)^2(\Delta_\alpha^2 + H_1(\alpha)\Delta_\alpha + H_0(\alpha)) \\ &= Q_{1,1}(\alpha)^2 H(\alpha, \Delta_\alpha). \end{aligned} \tag{15}$$

ii) Based on *Lemma 4*, for each $j \in \mathcal{E}\backslash\mathsf{S}[\mathcal{E}]$, $Q_{1,1}(\alpha_j) \neq 0$. In conjunction with $Q_1(P_j) = 0$, $\Delta_{\alpha_j} = \beta_j$. According to (15), $D(\alpha_j) = Q_{1,1}(\alpha_j)^2 H(\alpha_j, \beta_j) = 0$, which indicates $(x - \alpha_j) \mid D(x)$. For each $j \in \mathsf{S}[\mathcal{E}]$, $(x - \alpha_j) \mid Q_{i,1}(x)$ for $i = 0,1$. According to (13), we have $(x - \alpha_j)^2 \mid D(x)$. Therefore,

$$\prod_{j\in\mathcal{E}}(x - \alpha_j) = \prod_{j\in\mathcal{E}\backslash\mathsf{S}[\mathcal{E}]}(x - \alpha_j) \prod_{\alpha\in\{\alpha_j : j\in\mathsf{S}[\mathcal{E}]\}}(x - \alpha)^2$$

must be a factor for $D(x)$.

iii) Consider $Q_1(x,y) = \min \mathcal{M}_\mathcal{E}$. Assume that there exists $\alpha \in \mathbb{F}$ that satisfies $\alpha \neq \alpha_j$ for all $j \in \mathcal{E}$ and $D(\alpha) = 0$. Based on (15), if $Q_{1,1}(\alpha) \neq 0$, then $H(\alpha, \Delta_\alpha) = 0$, i.e., $(\alpha, \Delta_\alpha) \in \mathcal{P}$. Since $Q_1(\alpha, \Delta_\alpha) = 0$, there exists $j \in \mathcal{E}$ such that $\alpha_j = \alpha$. Based on (14), if $Q_{1,1}(\alpha) = 0$, then $Q_{0,1}(\alpha) = 0$, which leads to $Q_1(\alpha, y) = 0$, i.e., $(x - \alpha) \mid Q_1(x,y)$. Since $\alpha \neq \alpha_j$ for all $j \in \mathcal{E}$, $Q_1(x,y)/(x - \alpha)$ is a polynomial in $\mathcal{M}_\mathcal{E}$ with a $(2,b)$-weighted degree smaller than $\deg_{2,b} Q_1$. Therefore, the assumption leads to a contradiction. □

Corollary 6. *The determinant polynomial has the form of $D(x) = \tilde{D}(x)\prod_{j\in\mathcal{E}}(x - \alpha_j)$, where $\tilde{D}(x) \in \mathbb{F}[x]$ satisfies*

$$\deg \tilde{D} = \deg_{2,b} Q_1 - |\mathcal{E}|. \tag{16}$$

Consequently, the solution to (12) can be obtained by

$$(\hat{f}_0(x), \hat{f}_1(x)) = (\frac{N_0(x)}{D(x)}, \frac{N_1(x)}{D(x)}) \tag{17}$$

if it exists, where

$$\begin{aligned} N_0(x) &= -V_1(x)Q_{0,0}(x) + V_0(x)Q_{1,0}(x), \\ N_1(x) &= Q_{1,1}(x)Q_{0,0}(x) - Q_{0,1}(x)Q_{1,0}(x). \end{aligned} \tag{18}$$

Furthermore, *Lemma 7* reveals that $N_0(x)$ and $N_1(x)$ contain a factor of $\prod_{j\in\mathcal{E}}(x-\alpha_j)$ as well as $D(x)$. Its proof is similar to that of the property ii) of *Theorem 5*.

Lemma 7. *The polynomials $N_0(x)$ and $N_1(x)$ defined in (18) are divisible by $\prod_{j\in\mathcal{E}}(x-\alpha_j)$.*

IV. Determination of Codeword Symbols over Error Positions

With the error position set $\mathcal{E}$, this section shows how to determine the codeword symbols over $\mathcal{E}$ based on the root-finding equations over $\mathbb{F}[x]$. Since $D(x)$ has zeros at α_j for $j\in\mathcal{E}$, computing evaluations $\hat{f}_0(\alpha_j)$ and $\hat{f}_1(\alpha_j)$, and further the codeword symbol $\hat{f}(P_j)$ requires the Hasse derivative.

Theorem 8 *([26]). Assume that $h(x) = (x-\gamma)^{\mu}\tilde{h}(x)$, where $\gamma\in\mathbb{F}$, $\mu \geqslant 0$ and $\tilde{h}(x)\in\mathbb{F}[x]$. Let $h^{[\mu]}(x)$ denote the μth-order Hasse derivative of $h(x)$.*[4] *Then $\tilde{h}(\gamma) = h^{[\mu]}(\gamma)$.*

Let μ_j denote the zero order of $D(x)$ at α_j. If both $N_0(x)$ and $N_1(x)$ have a zero of order at least μ_j at α_j, then

$$\hat{f}_i(\alpha_j) = \frac{N_i^{[\mu_j]}(\alpha_j)}{D^{[\mu_j]}(\alpha_j)} \tag{19}$$

for $i=0,1$. Otherwise, equation (12) has no solution in $\mathbb{F}[x]$[2] and the decoder declares *failure*.

As shown in *Theorem 5* and *Lemma 7*, $D(x)$, $N_0(x)$ and $N_1(x)$ contain a factor of $\prod_{j\in\mathcal{E}}(x-\alpha_j)$. Hence, they all have a zero of order at least one at α_j for each $j\in\mathcal{E}$. In particular, for $j\in\mathsf{S}[\mathcal{E}]$, they all have a zero of order at least two at α_j, since $\mathsf{s}[j]\in\mathcal{E}$ and $\alpha_j = \alpha_{\mathsf{s}[j]}$. Let us define

$$\begin{aligned} \mathcal{E}^{\mathrm{I}} &= \{j\in\mathcal{E}\backslash\mathsf{S}[\mathcal{E}] : \mu_j = 1\}, \\ \mathcal{E}^{\mathrm{II}} &= \{j\in\mathsf{S}[\mathcal{E}] : \mu_j = 2\} \end{aligned}$$

and $\mathcal{E}^{\mathrm{III}} = \mathcal{E}\backslash(\mathcal{E}^{\mathrm{I}}\cup\mathcal{E}^{\mathrm{II}})$. Computing (19) for $j\in\mathcal{E}^{\mathrm{I}}\cup\mathcal{E}^{\mathrm{II}}$ involves only low-order Hasse derivatives. Furthermore, *Remark 9* reveals that $\mathcal{E}^{\mathrm{III}}$ contains relatively few error positions.

Remark 9. *Based on Corollary 6, $\mathcal{E}^{\mathrm{III}} = \{j\in\mathcal{E} : \tilde{D}(\alpha_j) = 0\}$. Since there may exist $j\in\mathsf{S}[\mathcal{E}]$ such that $\tilde{D}(\alpha_j) = 0$, $|\mathcal{E}^{\mathrm{III}}| \leqslant 2\deg\tilde{D}$. Along with (8) and (16), $|\mathcal{E}^{\mathrm{III}}| \leqslant 2g$.*

Consider $Q_1(x,y) = \min\mathcal{M}_{\mathcal{E}}$. Based on the property iii) of Theorem 5, $\mathcal{E}^{\mathrm{III}} \neq \emptyset$ if and only if $\deg\tilde{D} = 1$, or $\deg\tilde{D} > 1$ and $\tilde{D}(x)$ is reducible. In the case of $\deg\tilde{D} = 1$, we have $\mathcal{E}^{\mathrm{III}} = \{j^*\}$ where $j^*\in\mathcal{E}\backslash\mathsf{S}[\mathcal{E}]$ such that $\mu_{j^*} = 2$, or $\mathcal{E}^{\mathrm{III}} = \{j^*, \mathsf{s}[j^*]\}$ where $j^*\in\mathsf{S}[\mathcal{E}]$ such that $\mu_{j^*} = \mu_{\mathsf{s}[j^*]} = 3$.

Based on the above discussion, the computation of (19) can be categorized into three cases.

[4]Note that $h^{[0]}(x) = h(x)$

Case I: when $j\in\mathcal{E}\backslash\mathsf{S}[\mathcal{E}]$, based on equations (11), (13), (18) and the product rule of Hasse derivative [27, Chapter 1],

$$\begin{aligned} D^{[1]}(\alpha_j) &= \sum_{i=0}^{1}\sum_{\mu=0}^{1}(1-2i)V_{1-i}^{[1-\mu]}(\alpha_j)Q_{i,1}^{[\mu]}(\alpha_j), \\ N_0^{[1]}(\alpha_j) &= \sum_{i=0}^{1}\sum_{\mu=0}^{1}(2i-1)V_{1-i}^{[1-\mu]}(\alpha_j)Q_{i,0}^{[\mu]}(\alpha_j), \\ N_1^{[1]}(\alpha_j) &= \sum_{i=0}^{1}\sum_{\mu=0}^{1}(1-2i)Q_{1-i,1}^{[1-\mu]}(\alpha_j)Q_{i,0}^{[\mu]}(\alpha_j), \end{aligned} \tag{20}$$

where

$$\begin{aligned} V_0^{[1]}(\alpha_j) &= -H_0^{[1]}(\alpha_j)Q_{1,1}(\alpha_j) - H_0(\alpha_j)Q_{1,1}^{[1]}(\alpha_j), \\ V_1^{[1]}(\alpha_j) &= Q_{0,1}^{[1]}(\alpha_j) - H_1^{[1]}(\alpha_j)Q_{1,1}(\alpha_j) - H_1(\alpha_j)Q_{1,1}^{[1]}(\alpha_j). \end{aligned}$$

Based on *Theorem 8*, $j\in\mathcal{E}^{\mathrm{I}}$ if and only if $D^{[1]}(\alpha_j)\neq 0$.

Case II: when $j\in\mathsf{S}[\mathcal{E}]$, $D^{[2]}(\alpha_j)$, $N_0^{[2]}(\alpha_j)$ and $N_1^{[2]}(\alpha_j)$ can also be derived from the first-order Hasse derivatives as

$$\begin{aligned} D^{[2]}(\alpha_j) &= \tilde{V}_1(\alpha_j)Q_{0,1}^{[1]}(\alpha_j) - \tilde{V}_0(\alpha_j)Q_{1,1}^{[1]}(\alpha_j), \\ N_0^{[2]}(\alpha_j) &= -\tilde{V}_1(\alpha_j)Q_{0,0}^{[1]}(\alpha_j) + \tilde{V}_0(\alpha_j)Q_{1,0}^{[1]}(\alpha_j), \\ N_1^{[2]}(\alpha_j) &= Q_{1,1}^{[1]}(\alpha_j)Q_{0,0}^{[1]}(\alpha_j) - Q_{0,1}^{[1]}(\alpha_j)Q_{1,0}^{[1]}(\alpha_j), \end{aligned} \tag{21}$$

where

$$\begin{aligned} \tilde{V}_0(\alpha_j) &= -H_0(\alpha_j)Q_{1,1}^{[1]}(\alpha_j), \\ \tilde{V}_1(\alpha_j) &= Q_{0,1}^{[1]}(\alpha_j) - H_1(\alpha_j)Q_{1,1}^{[1]}(\alpha_j). \end{aligned}$$

Based on *Theorem 8*, $j\in\mathcal{E}^{\mathrm{II}}$ if and only if $D^{[2]}(\alpha_j)\neq 0$.

Case III: when $j\in\mathcal{E}^{\mathrm{III}}$, computation of (19) involves the Hasse derivatives of an order greater than one. As an alternative, we can determine $\hat{f}(P_j)$ for all $j\in\mathcal{E}^{\mathrm{III}}$ using general erasure decoding, e.g., solving the parity-check equations. If $u < n - |\mathcal{E}^{\mathrm{III}}|$, any $|\mathcal{E}^{\mathrm{III}}|$ erasures can be corrected with at most $|\mathcal{E}^{\mathrm{III}}| + g$ parity-check equations. As discussed in *Remark 9*, $|\mathcal{E}^{\mathrm{III}}| \leqslant 2\deg\tilde{D} \leqslant 2g$. In fact, it is shown in Section V that $\mathcal{E}^{\mathrm{III}}$ is often empty or small. Its cardinality struggles to approach $2g$ as g grows.

When the ReT [21] [22] is applied to the interpolation-based decoding, there are minor modifications to the computations of $Q_{i,0}(\alpha_j)$ and $Q_{i,0}^{[1]}(\alpha_j)$ for $i=0,1$.

Remark 10. *Assume that $r_j = 0$ for all $j\in\mathcal{R}$, where $\mathcal{R}$ is a set of $2\lfloor(k-g)/2\rfloor$ positions satisfying $\mathcal{R} = \{j\in\mathcal{R} : \mathsf{s}[j]\in\mathcal{R}\}$. Let $\phi(x) = \prod_{\alpha\in\{\alpha_j : j\in\mathcal{R}\}}(x-\alpha)$. It is shown in [21] [22] that $\phi(x)$ is a factor for $Q_0(x,y)$, i.e.,*

$$Q_0(x,y) = \phi(x)\mathcal{Q}_0(x,y), \tag{22}$$

where $\mathcal{Q}_0(x,y) = \mathcal{Q}_{0,0}(x) + \mathcal{Q}_{1,0}(x)y$ and $Q_{i,0}(x) = \phi(x)\mathcal{Q}_{i,0}(x)$ for $i=0,1$. With the ReT, the interpolation outcome is $\mathcal{Q}(x,y,z) = \mathcal{Q}_0(x,y) + Q_1(x,y)z$. Based on the product rule of Hasse derivative, $Q_{i,0}(\alpha_j)$ and $Q_{i,0}^{[1]}(\alpha_j)$ can be computed by

$$\begin{aligned} Q_{i,0}(\alpha_j) &= \phi(\alpha_j)\mathcal{Q}_{i,0}(\alpha_j), \\ Q_{i,0}^{[1]}(\alpha_j) &= \phi^{[1]}(\alpha_j)\mathcal{Q}_{i,0}(\alpha_j) + \phi(\alpha_j)\mathcal{Q}_{i,0}^{[1]}(\alpha_j), \end{aligned}$$

where $\phi(\alpha_j)$ and $\phi^{[1]}(\alpha_j)$ can be pre-computed and $\phi(\alpha_j) = 0$ for all $j \in \mathcal{R}$. It is shown that the restoration of $Q_0(x, y)$ as in (22) *is unnecessary for the proposed root-finding algorithm.*

The following *Remark 11* provides an upper bound on the complexity of determining $\hat{f}(P_j)$ for all $j \in \mathcal{E}$, which is obtained by assuming that $\mathcal{E} = \mathcal{E}^{\mathrm{I}}$. The rationale behind the assumption is that the computation of **Case I** is more complex than that of **Case II**, and $|\mathcal{E}^{\mathrm{III}}|$ is often small.

Remark 11. *For $i = 0, 1$ and $j = 1, 2, ..., n$, $H_i(\alpha_j)$ and $H_i^{[1]}(\alpha_j)$ can be pre-computed, and $Q_{i,1}(\alpha_j)$ have been obtained in the determination of $\mathcal{E}$. Hence, computing $Q_{i,0}(\alpha_j)$, $Q_{i,0}^{[1]}(\alpha_j)$ and $Q_{i,1}^{[1]}(\alpha_j)$ for $i = 0, 1$ and $j \in \mathcal{E}$ dominates the complexity. Based on* (8) *and* (9)*, it requires less than*

$$(2\deg_{2,b} Q_0 + \deg_{2,b} Q_1)|\mathcal{E}| < 3|\mathcal{E}|^2 + 2k|\mathcal{E}| + 5g|\mathcal{E}|$$

multiplications. Furthermore, with the ReT, the number of multiplications can be reduced to less than

$$3|\mathcal{E}|^2 + 7g|\mathcal{E}|.$$

This is due to $\deg_{2,b} \mathcal{Q}_0 = \deg_{2,b} Q_0 - 2\lfloor (k-g)/2 \rfloor$. The actual number may be even smaller, because for $j \in \mathcal{R} \cap \mathcal{E}$, $Q_{i,0}(\alpha_j) = 0$ and $Q_{i,0}^{[1]}(\alpha_j) = \phi^{[1]}(\alpha_j) Q_{i,0}(\alpha_j)$.

V. Numerical Results

This section presents some numerical results to demonstrate the effectiveness of the necessary conditions (8) and (9) in filtering invalid decodings, the limited computational cost of **Case III**, and finally the complexity advantage of the proposed algorithm over the RR algorithm. Since the comparison is carried out at moderate codeword lengths, the RR algorithm is implemented without divide-and-conquer extensions. The $(80, 41)$ elliptic code and the $(128, 96)$ hyperelliptic code are implemented for the decoding. They are constructed from $Y^2 = Y + X^3$ and $Y^2 = Y + X^9$ over GF(64) with a designed distance of 39 and 29, respectively.

TABLE I
Decoding Statistics of the $(80, 41)$ Elliptic Code and the $(128, 96)$ Hyperelliptic Code

$(n,\ k)$	(80, 41)		(128, 96)			
#error	19	20	13	14	15	16
(8) & (9)	100,000	1,197	100,000	100,000	1,526	23
#success	100,000	1,197	100,000	100,000	1,526	23

Table I shows the decoding statistics for the two codes w.r.t. different numbers of errors. Each column shows the statistics of the instances where (8) and (9) are met, and the instances of successful decoding, under the specified number of errors. They were obtained by decoding the received words of 100,000 randomly generated codewords. It shows that the decoding always succeeds when (8) and (9) are satisfied, demonstrating their effectiveness in filtering invalid decodings. It can also be seen that when the number of errors does not exceed $\lfloor (d^* - 1)/2 \rfloor$, the decoding always succeeds though the theoretical error-correction radius is $\lfloor (d^* - g - 1)/2 \rfloor$.

TABLE II
Statistics of $|\mathcal{E}^{\mathrm{III}}|$ in Decoding the $(128, 96)$ Hyperelliptic Code

#error	12	13	14	15	16
#success	100,000	100,000	100,000	1,526	23
$\lvert\mathcal{E}^{\mathrm{III}}\rvert = 0$	80,797	79,453	78,728	1,183	18
$\lvert\mathcal{E}^{\mathrm{III}}\rvert = 1$	15,883	16,752	17,021	270	3
$\lvert\mathcal{E}^{\mathrm{III}}\rvert = 2$	2,989	3,359	3,722	67	2
$\lvert\mathcal{E}^{\mathrm{III}}\rvert = 3, 4, 5$	331	436	529	6	0
$\lvert\mathcal{E}^{\mathrm{III}}\rvert \geqslant 6$	0	0	0	0	0

Table II shows the statistics of $|\mathcal{E}^{\mathrm{III}}|$ in decoding the $(128, 96)$ hyperelliptic code. It shows that when the decoding proceeds to determine the codeword symbols, $\mathcal{E}^{\mathrm{III}}$ often remains empty or small. This indicates that **Case III** will only introduce limited computational cost.

TABLE III
Average finite field multiplications in Decoding the $(80, 41)$ Elliptic Code and the $(128, 96)$ Hyperelliptic Code

$(n,\ k)$	(80, 41)			(128, 96)		
#error	18	19	20	13	14	15
RR*	3,281	3,343	1,675	10,973	11,114	4,955
Proposed*	1,560	1,661	797	1,516	1,663	954
RR	2,542	2,583	948	8,031	8,126	2,054
Proposed	2,462	2,605	806	3,450	3,699	1,008

* : With the ReT.

Finally, Table III compares the average complexity of the proposed algorithm and the RR algorithm. The complexity is measured as the average number of finite field multiplications in decoding a codeword. In the RR algorithm, the estimated message is encoded to obtain the estimated codeword. It can be seen that with the ReT, the proposed algorithm exhibits a significant advantage over the RR algorithm, since the latter requires a restoration of the interpolation polynomial. Without the ReT, the proposed algorithm outperforms the RR algorithm in decoding the $(128, 96)$ hyperelliptic code, while they have a similar complexity in decoding the $(80, 41)$ elliptic code. This indicates that the proposed algorithm is more effective for high rate codes.

It should be pointed out that the proposed algorithm can achieve a higher degree of computational parallelism over the RR algorithm. Its computation mainly consists of polynomial evaluations, which can be performed in parallel. This feature will be welcomed by practical implementations.

Acknowledgement

This work is sponsored by the National Natural Science Foundation of China (NSFC) with project ID 62071498. The authors would like to thank the anonymous reviewers for their helpful remarks.

REFERENCES

[1] V. D. Goppa, "Algebraico-geometric codes," *Izv. Akad. Nauk SSSR Ser. Mat.*, vol. 46, no. 4, pp. 762–781, 1982.

[2] S. Miura and N. Kamiya, "Geometric-Goppa codes on some maximal curves and their minimum distance," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Jun. 1993, pp. 85–86.

[3] J. Justesen, K. Larsen, H. Jensen and *et al.*, "Construction and decoding of a class of algebraic geometry codes," *IEEE Trans. Inf. Theory*, vol. 35, no. 4, pp. 811–821, Jul. 1989.

[4] G. Feng and T. R. Rao, "Decoding algebraic-geometric codes up to the designed minimum distance," *IEEE Trans. Inf. Theory*, vol. 39, no. 1, pp. 37–45, Jan. 1993.

[5] S. Sakata, H. Jensen, and T. Hoholdt, "Generalized Berlekamp-Massey decoding of algebraic-geometric codes up to half the Feng-Rao bound," *IEEE Trans. Inf. Theory*, vol. 41, no. 6, pp. 1762–1768, 1995.

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

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

[8] D. Leonard, "A generalized Forney formula for algebraic-geometric codes," *IEEE Trans. Inf. Theory*, vol. 42, no. 4, pp. 1263–1268, Jul. 1996.

[9] J. Hansen, H. Jensen, and R. Kötter, "Determination of error values for algebraic-geometry codes and the Forney formula," *IEEE Trans. Inf. Theory*, vol. 44, no. 5, pp. 1881–1886, 1998.

[10] D. Leonard, "Efficient Forney functions for decoding AG codes," *IEEE Trans. Inf. Theory*, vol. 45, no. 1, pp. 260–265, Jan. 1999.

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

[12] P. Beelen and M. Montanucci, "List-decoding of AG codes without genus penalty," *preprint available as arXiv.2307.04203*, Jul. 2023.

[13] 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.

[14] 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.

[15] Y. Wan, 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.

[16] 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.

[17] 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, 2005.

[18] 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.

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

[20] R. Kötter, 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.

[21] 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.

[22] Y. Wan, J. Xing, Y. Huang and *et al.*, "The re-encoding transform in algebraic list decoding of algebraic geometric codes," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Taipei, Taiwan, Jun. 2023, pp. 19–24.

[23] H. Stichtenoth, *Algebraic Function Fields and Codes*, 2nd ed. Berlin: Springer, 2009, no. 254.

[24] S. Miura, "Algebraic geometric codes on certain plane curves," *Electron. Commun. Jpn. III, Fundam. Electron. Sci.*, vol. 76, no. 12, pp. 1–13, 1993.

[25] P. Beelen and T. Høholdt, "The decoding of algebraic geometry codes," in *Advances in Algebraic Geometry Codes*, E. Martínez-Moro, C. Munera, and D. Ruano, Eds. Singapore: World Scientific, 2008, vol. 5, pp. 49–98.

[26] H. Hasse, "Theorie der höheren differentiale in einem algebraischen funktionenkörper mit vollkommenem konstantenkörper bei beliebiger charakteristik." *Journal für die reine und angewandte Mathematik*, vol. 1936, no. 175, pp. 50–54, 1936.

[27] D. M. Goldschmidt, *Algebraic Functions and Projective Curves*. New York: Springer, 2003, vol. 215.