Systematic Encoding of Elliptic Codes with Dynamic Information Sets

Jianguo Zhao†, 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
Email: zhaojg5@mail2.sysu.edu.cn, chenli55@mail.sysu.edu.cn

2023 IEEE International Symposium on Information Theory (ISIT) | 978-1-6654-7554-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/ISIT54713.2023.10206531

***Abstract*—Systematic encoding is crucial in some decoding algorithms of algebraic-geometric (AG) codes. In the context of soft decoding that requires systematic encoding, the information set changes dynamically for each decoding event, presenting a challenge for systematic encoding of AG codes. This paper addresses this challenge for elliptic codes by proposing a fast construction algorithm for both the systematic encoding polynomial (SEP) and the systematic generator matrix (SGM) based on Gröbner bases of modules and backward interpolation. The proposed algorithm has a complexity of $\mathcal{O}(k^2)$ and $\mathcal{O}(kn)$ for constructing the SEP and SGM, respectively, where k and n denote the dimension and length of the elliptic code. Additionally, its computation can be highly parallelized, enhancing its implementation efficiency.**

***Index Terms*—Algebraic-geometric codes, backward interpolation, elliptic codes, Gröbner bases, systematic encoding**

I. Introduction

Algebraic-geometric (AG) codes are a generalization of the widely-used Reed-Solomon (RS) codes. Constructed from curves of genus one, elliptic codes are a class of AG codes that are almost maximum distance separable (MDS) and can be longer than RS codes. Therefore, they offer a good tradeoff between length and distance, which contributes to their potential to replace RS codes in future applications. Decoding of AG codes includes syndrome-based [1]–[3] and interpolation-based approaches [4]. Additionally, information set decoding for general linear codes can also be applied.

In some decoding algorithms, systematic encoding plays an important role. For example, erasure-only decoding can be realized through systematic encoding with the unerased positions [5] [6]. The re-encoding transform (ReT) [7] constructs a systematic encoding polynomial (SEP) based on the most reliable received symbols, transforming the received word and significantly reducing the complexity of interpolation-based decoding [7]–[10]. Moreover, systematic encoding is indispensable for information set decoding. For example, the ordered statistics decoding (OSD) [11] utilizes a systematic generator matrix (SGM) defined by the most reliable independent positions (MRIPs) to generate codeword candidates.

In the context of soft decoding that requires systematic encoding, the positions of information symbols depend on the reliabilities of received symbols. As a result, the information set changes dynamically for each decoding event, presenting a challenge for constructing the SEP or SGM for AG codes. Although general linear algebra provides Gaussian elimination (GE) as a solution, it is inefficient due to its cubic complexity and sequential operation.

Few studies have addressed systematic encoding of AG codes [6] [12] [13], and most of them did not consider the dynamic information set. The ReT for elliptic codes [8] [14] was proposed by using the Lagrange interpolation, where the interpolation points associated with the information set are required to form a semi-grid (*Definition III*). However, this requirement cannot always be met by the information set consisting of MRIPs. An algorithm for inverting the encoding map [15] was introduced for certain AG codes. It starts with the Lagrange interpolation and further reduces the interpolation polynomial to a message polynomial through multivariate division by a Gröbner basis. Although not specified, it can lead to an SEP construction algorithm.

This paper proposes a fast construction algorithm for the SEP and SGM of elliptic codes with dynamic information sets. It is shown that both the constructions can be reduced to determining a set of Gröbner bases. Given an information set J_k of an (n, k) elliptic code, the proposed algorithm first computes a Gröbner basis $\mathcal{G}_{J_k}$ of the constrained submodule $\mathcal{M}_{J_k}$ associated with J_k. Subsequently, the backward interpolation [16] [17] is used to derive the desired Gröbner bases from $\mathcal{G}_{J_k}$. The proposed algorithm yields a complexity of $\mathcal{O}(k^2)$ and $\mathcal{O}(kn)$ for constructing the SEP and SGM, respectively. Its computation can be highly parallelized, exhibiting an advantageous implementation feature.

II. Preliminaries

A. Elliptic Codes

Over finite field $\mathbb{F}_q$, an affine elliptic curve $\mathcal{X}$ is defined by the following equation

$$\mathcal{X} : Y^2 + \gamma_1 XY + \gamma_3 Y = X^3 + \gamma_2 X^2 + \gamma_4 X + \gamma_6, \quad (1)$$

where $\gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_6 \in \mathbb{F}_q$. The genus of $\mathcal{X}$ is 1. Let $\mathcal{X}(\mathbb{F}_q)$ denote the set of rational points on $\mathcal{X}$. Based on the Hasse-Weil bound, $|\mathcal{X}(\mathbb{F}_q)| \leqslant q + \lfloor 2\sqrt{q} \rfloor + 1$. The rational points consist of a set of affine points $\mathcal{P} = \{P_j = (\alpha_j, \beta_j) : j = 1, 2, ..., n\}$ and a point at infinity P_∞, i.e., $\mathcal{X}(\mathbb{F}_q) = \mathcal{P} \cup \{P_\infty\}$. They form an additive Abelian group with P_∞ as the identity element [18]. For any $P_j \in \mathcal{P}$, there exists $P_{s[j]} = (\alpha_j, -\beta_j - \gamma_1\alpha_j - \gamma_3) \in \mathcal{P}$, s.t. $P_j + P_{s[j]} = P_\infty$ under the "chord-and-

tangent" rule. Note that P_j and $P_{\mathrm{s}[j]}$ [1] are the only points with the same X-coordinate in $\mathcal{P}$.

Let $\mathbb{F}_q(\mathcal{X})$ denote the function field of $\mathcal{X}$. For any $a, b \geqslant 0$, $x^a y^b \in \mathbb{F}_q(\mathcal{X})$ has poles only at P_∞ with an order of $-2a-3b$. Furthermore, any $f = \sum_{a\geqslant 0}\sum_{b\geqslant 0} f_{a,b}x^a y^b \in \mathbb{F}_q[x,y]\backslash\{0\}$ has poles only at P_∞ and the pole order is equal to its $(2,3)$-weighted degree $\deg_{2,3} f = \max_{a,b}\{2a+3b : f_{a,b} \neq 0\}$. Based on (1), every $f \in \mathbb{F}_q[x,y]$ with y-degree $\deg_y f \geqslant 2$ can be uniquely rewritten as another bivariate polynomial $f' \in \mathbb{F}_q[x,y]$ with $\deg_y f' < 2$. Hence, $\mathbb{F}_q[x,y]$ is equivalent to the following set:

$$\begin{aligned}\mathbb{F}_q[x][y]_2 =&\{f(x,y) \in \mathbb{F}_q[x,y] : \deg_y f < 2\}\\ =&\{f_{[0]}(x) + f_{[1]}(x)y : f_{[0]}, f_{[1]} \in \mathbb{F}_q[x]\},\end{aligned}$$

which is a free module over $\mathbb{F}_q[x]$ with basis $\{1, y\}$.

Armed with the above prerequisites, the Riemann-Roch space associated with divisor $k[P_\infty]$ can be represented as

$$\mathcal{L}(k[P_\infty]) = \{f \in \mathbb{F}_q[x][y]_2 : \deg_{2,3} f \leqslant k\}. \tag{2}$$

This paper only considers $0 < k < n$, for which $\mathcal{L}(k[P_\infty])$ has a dimension of k over $\mathbb{F}_q$.

Definition I *([19]).* An (n,k) (one-point evaluation) elliptic code $\mathbb{C}_{\mathcal{L}}(\mathcal{P},k)$ is defined as the image of $\mathcal{L}(k[P_\infty])$ under the evaluation map

$$\begin{aligned}\mathrm{E} : \mathcal{L}(k[P_\infty]) &\to \mathbb{F}_q^n\\ f &\mapsto (f(P_1), f(P_2), ..., f(P_n)).\end{aligned}$$

Note that the minimum distance d of $\mathbb{C}_{\mathcal{L}}(\mathcal{P},k)$ satisfies $d \geqslant n-k$, and the map E is bijective for $0 < k < n$.

B. Gröbner Bases of Modules

The Gröbner basis theory can be used to solve the following multivariate polynomial interpolation problem in module $\mathbb{F}_q[x][y]_2$.

Problem 1. Given a set of codeword symbol postitions $J \subseteq \{1, 2, ..., n\}$, find a nonzero polynomial $Q \in \mathbb{F}_q[x][y]_2$, s.t. $Q(P_j) = 0, \forall j \in J$ and $\deg_{2,3} Q$ is minimal.

Let $\mathcal{P}_J = \{P_j : j \in J\}$ denote the set of interpolation points in *Problem 1*. It defines a submodule of $\mathbb{F}_q[x][y]_2$ as

$$\mathcal{M}_J = \{Q \in \mathbb{F}_q[x][y]_2 : Q(P_j) = 0, \forall j \in J\}.$$

By ordering polynomials in $\mathbb{F}_q[x][y]_2$ according to their $(2,3)$-weighted degree, the solution to *Problem 1* will be the minimal element of $\mathcal{M}_J$, which is contained in a Gröbner basis of $\mathcal{M}_J$.

Definition II. Given $f, f' \in \mathbb{F}_q[x][y]_2$, $f < f'$ w.r.t. order $<_{2,3}$ if $\deg_{2,3} f < \deg_{2,3} f'$.

Since 2 and 3 are coprime, any $f \in \mathbb{F}_q[x][y]_2$ has a well-defined leading monomial $\mathrm{LM}_{2,3} f = \max_{a,b}\{x^a y^b : f_{a,b} \neq 0\}$ under $<_{2,3}$. Then, the Gröbner basis of $\mathcal{M}_J$ w.r.t. $<_{2,3}$ can be determined using the following criterion.

Proposition 1 *([20]).* A polynomial set $\mathcal{G}$ is a Gröbner basis of $\mathcal{M}_J$ w.r.t. $<_{2,3}$ iff it satisfies:

1) $\mathcal{G} = \{Q^{(0)}, Q^{(1)}\}$;
2) $\mathcal{M}_J = \langle\mathcal{G}\rangle := \{h_0 Q^{(0)} + h_1 Q^{(1)} : h_0, h_1 \in \mathbb{F}_q[x]\}$;
3) $\deg_y \mathrm{LM}_{2,3} Q^{(0)} \neq \deg_y \mathrm{LM}_{2,3} Q^{(1)}$.

Let $\mathcal{G}_J$ denote the Gröbner basis of $\mathcal{M}_J$. Under the order $<_{2,3}$, the minimal element of $\mathcal{M}_J$ is equivalent to that of $\mathcal{G}_J$. Therefore, *Problem 1* can be solved by computing $\mathcal{G}_J$. To achieve this, Kötter's interpolation [21] can be utilized. It first initializes a Gröbner basis $\{1, y\}$ of the unconstrained module $\mathbb{F}_q[x][y]_2$. Then, the interpolation constraints given by $\mathcal{P}_J$ are imposed on the Gröbner basis one by one using *Algorithm 1*, where $\mathrm{F}_{\Delta^{(0)},\Delta^{(1)}}$ is a map defined as

$$\mathrm{F}_{\delta_0,\delta_1} : \mathbb{F}_q[x][y]_2 \times \mathbb{F}_q[x][y]_2 \to \mathbb{F}_q[x][y]_2$$

$$\mathrm{F}_{\delta_0,\delta_1}(\varphi_0,\varphi_1) = \begin{cases}\varphi_0, \text{ if } \delta_0 = 0 \text{ and } \delta_1 \neq 0\\ \varphi_1, \text{ if } \delta_0 \neq 0 \text{ and } \delta_1 = 0\\ \varphi_0 - \frac{\delta_0}{\delta_1}\varphi_1, \text{ if } \delta_0 \neq 0 \text{ and } \delta_1 \neq 0\end{cases}$$

for $\delta_0, \delta_1 \in \mathbb{F}_q$ that are not all zeros.

Algorithm 1: Kötter's Basis Update (**`BasisUpdate`**)

Input: $\mathcal{G} = \{Q^{(0)}, Q^{(1)}\}, P = (\alpha, \beta) \in \mathcal{P}$
Output: $\mathcal{G}$

for $l = 0, 1$ **do**
  $\Delta^{(l)} \leftarrow Q^{(l)}(P)$;
if $\Delta^{(l)} = 0, \forall l$ **then** stop;
$l^* \leftarrow \arg\min_l\{Q^{(l)} : \Delta^{(l)} \neq 0\}$;
$\mathcal{G} \leftarrow \{\mathrm{F}_{\Delta^{(0)},\Delta^{(1)}}(Q^{(0)}, Q^{(1)}), (x-\alpha)Q^{(l^*)}\}$;

Based on *Algorithm 1*, Kötter's interpolation can be summarized as follows.

Algorithm 2: Kötter's Interpolation (**`KötterInterp`**)

Input: $\mathcal{P}, J$
Output: $\mathcal{G}$

$\mathcal{G} \leftarrow \{1, y\}$;
for $j \in J$ **do**
  $\mathcal{G} \leftarrow$ **`BasisUpdate`**$(\mathcal{G}, P_j)$;

The following *Lemma 2* characterizes a bound on the $(2,3)$-weighted degree for the elements of $\mathcal{G}_J$.

Lemma 2. Let $\mathcal{G}_J^{(-)}$ and $\mathcal{G}_J^{(+)}$ denote the minimal and maximal element of $\mathcal{G}_J$ w.r.t. $<_{2,3}$, respectively. Their $(2,3)$-weighted degree satisfy i) $|J| \leqslant \deg_{2,3} \mathcal{G}_J^{(-)} \leqslant |J|+1$ and ii) $\deg_{2,3} \mathcal{G}_J^{(+)} \leqslant \deg_{2,3} \mathcal{G}_J^{(-)} + 3$. [2]

III. Problem Formulation

Given a message $\underline{u} = (u_1, u_2, ..., u_k) \in \mathbb{F}_q^k$, the systematic encoder of $\mathbb{C}_{\mathcal{L}}(\mathcal{P},k)$ with information set $J_k = \{j_1, j_2, ..., j_k\}$ generates a codeword $\underline{c} = (c_1, c_2, ..., c_n)$, s.t. $c_j = u_{\mathrm{i}[j]}, \forall j \in J_k$, where $\mathrm{i}[j]$ denotes the index of j in J_k, i.e., $\mathrm{i}[j_i] = i$ for $i = 1, 2, .., k$. Since the evaluation map E in *Definition I* is bijective for $0 < k < n$, there exists a unique message polynomial $\mathcal{F} \in \mathcal{L}(k[P_\infty])$ called the SEP, s.t. $\mathrm{E}(\mathcal{F}) = \underline{c}$. Systematic encoding of $\mathbb{C}_{\mathcal{L}}(\mathcal{P},k)$ can be realized through finding the SEP $\mathcal{F}$ and computing the evaluation image $\mathrm{E}(\mathcal{F})$ of $\mathcal{F}$.

[1] There may exist $j \in \{1, 2, ..., n\}$, s.t. $\mathrm{s}[j] = j$.

[2] Due to space limitations, the proofs of this lemma and any subsequent propositions are omitted.

Problem 2. Given an information set J_k of $\mathbb{C}_{\mathcal{L}}(\mathcal{P},k)$ and a message $\underline{u} \in \mathbb{F}_q^k$, find a polynomial $\mathcal{F} \in \mathbb{F}_q[x][y]_2$, s.t. $\mathcal{F}(P_j) = u_{\mathrm{i}[j]}, \forall j \in J_k$ and $\deg_{2,3} \mathcal{F} \leqslant k$.

Since $\mathcal{L}(kP_\infty)$ has dimension k over $\mathbb{F}_q$, $\mathcal{F}$ can be represented as an $\mathbb{F}_q$-linear combination of k basis polynomials of $\mathcal{L}(kP_\infty)$. Particularly, there exists a set of systematic encoding basis polynomials (SEBPs) $\{\mathcal{Q}_j \in \mathcal{L}(kP_\infty) : j \in J_k\}$, s.t.

$$\mathcal{F} = \sum_{j \in J_k} u_{\mathrm{i}[j]} \mathcal{Q}_j. \tag{3}$$

Hence, *Problem 2* can be solved by finding these SEBPs.

Problem 3. Given an information set J_k of $\mathbb{C}_{\mathcal{L}}(\mathcal{P},k)$, find polynomials $\mathcal{Q}_j \in \mathbb{F}_q[x][y]_2$ for all $j \in J_k$, s.t.

$$\begin{cases} \mathcal{Q}_j(P_j) = 1, \\ \mathcal{Q}_j(P_{j'}) = 0, \forall j' \in J_k \backslash \{j\}, \end{cases} \tag{4}$$

and $\deg_{2,3} \mathcal{Q}_j \leqslant k$.

Note that each $\mathcal{Q}_j$ is a solution to the special case of *Problem 2* with $u_{\mathrm{i}[j]} = 1$ and $u_{\mathrm{i}[j']} = 0, \forall j' \in J_k \backslash \{j\}$. The evaluation images $\mathrm{E}(\mathcal{Q}_{j_1}), ..., \mathrm{E}(\mathcal{Q}_{j_k})$ form the SGM $\mathbf{G}_{J_k}$ of $\mathbb{C}_{\mathcal{L}}(\mathcal{P},k)$ with the information set J_k, where $\mathbf{G}_{J_k} = (g_{\mathrm{i}[j],i})$ and $g_{\mathrm{i}[j],i} = \mathcal{Q}_j(P_i)$ for $j \in J_k$ and $i \in \{1,2,...,n\}$.

Due to its uniqueness, $\mathcal{Q}_j$ is the minimal polynomial that satisfies (4). In addition, for any $\gamma \in \mathbb{F}_q \backslash \{0\}$, $Q_j = \gamma \mathcal{Q}_j$ has $\deg_{2,3} Q_j = \deg_{2,3} \mathcal{Q}_j$. Hence, $\mathcal{Q}_j$ can be obtained by finding Q_j and computing $\mathcal{Q}_j = Q_j/\gamma = Q_j/Q_j(P_j)$.

Problem 4. Given an information set J_k of $\mathbb{C}_{\mathcal{L}}(\mathcal{P},k)$, find polynomials $Q_j \in \mathbb{F}_q[x][y]_2$ for all $j \in J_k$, s.t. $Q_j(P_{j'}) = 0, \forall j' \in J_k \backslash \{j\}$ and $\deg_{2,3} Q_j$ is minimal.

It can be observed that *Problem 4* comprises k interpolation problems defined by *Problem 1*. Each problem differs by one interpolation point. Thus, *Problem 4* can be solved by computing the k Gröbner bases $\mathcal{G}_{J_k \backslash \{j_1\}}, ..., \mathcal{G}_{J_k \backslash \{j_k\}}$. Consequently, $Q_j = \mathcal{G}^{(-)}_{J_k \backslash \{j\}}, \forall j \in J_k$.

IV. Fast Constructions of SEP and SGM

Section III demonstrates that both the SEP and SGM of $\mathbb{C}_{\mathcal{L}}(\mathcal{P},k)$ can be derived from the solution of *Problem 4*, which is contained in the Gröbner bases $\mathcal{G}_{J_k \backslash \{j_1\}}, ..., \mathcal{G}_{J_k \backslash \{j_k\}}$. Although applying *Algorithm 2* multiple times can yield the desired Gröbner bases, it results in redundant computation. This section shows that this redundancy can be eliminated by exploiting the similarity between these Gröbner bases. Figure 1 illustrates the fast SEP construction, including two key steps:

1) Forward computation: compute a Gröbner basis $\mathcal{G}_{J_k}$ of the module $\mathcal{M}_{J_k}$;
2) Backward computation: utilize the backward interpolation to generate the Gröbner basis $\mathcal{G}_{J_k \backslash \{j\}}$ from $\mathcal{G}_{J_k}$ and further derive the SEBP $\mathcal{Q}_j$ for all $j \in J_k$.

A. Backward Interpolation

The backward interpolation [16] [17] eliminates interpolation constraints from an existing Gröbner basis, yielding a new Gröbner basis. In this work, it is used to eliminate the interpolation constraint defined by P_j from $\mathcal{G}_{J_k}$ to generate $\mathcal{G}_{J_k \backslash \{j\}}$ for all $j \in J_k$. The problem of backward interpolation is formulated in *Problem 5*.

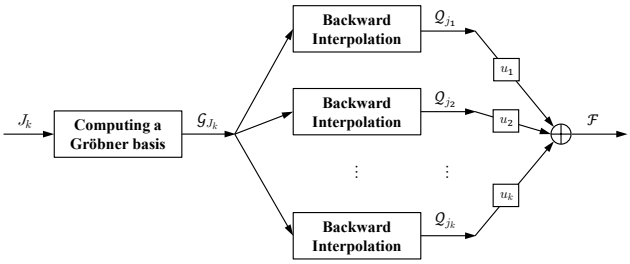

Fig. 1. Fast SEP construction.

Problem 5. Given a position set $J \subseteq \{1,2,...,n\}$, a Gröbner basis $\mathcal{G}_J$ of the module $\mathcal{M}_J$ and a position $j \in J$, compute a Gröbner basis $\mathcal{G}_{J \backslash \{j\}}$ of the module $\mathcal{M}_{J \backslash \{j\}}$.

If $Q \in \mathcal{M}_J$ is divisible by $x - \alpha_j$, then the zeros of Q at P_j can be reduced through dividing by $x - \alpha_j$, which means the constraint given by P_j can be eliminated. This provides a foundation for solving *Problem 5*.

Proposition 3. Given $\mathcal{G}_J = \{Q^{(0)}, Q^{(1)}\}$ and $j \in J$, let $L = \{l \in \{0,1\} : (x - \alpha_j) \mid Q^{(l)}\}$ and $L^c = \{0,1\} \backslash L$. If $L \neq \emptyset$, let $\tilde{Q}^{(l)} = Q^{(l)}/(x - \alpha_j)$ for $l \in L$. Then, $\mathcal{G}_{J \backslash \{j,\mathrm{s}[j]\}} = \{\tilde{Q}^{(l)} : l \in L\} \cup \{Q^{(l)} : l \in L^c\}$.

Remark 4. If $\mathrm{s}[j] \notin J$ or $\mathrm{s}[j] = j$, then $\mathcal{G}_{J \backslash \{j,\mathrm{s}[j]\}} = \mathcal{G}_{J \backslash \{j\}}$ is the solution to *Problem 5*. Otherwise, $\mathcal{G}_{J \backslash \{j,\mathrm{s}[j]\}} \neq \mathcal{G}_{J \backslash \{j\}}$ and it can be updated to $\mathcal{G}_{J \backslash \{j\}}$ using *Algorithm 1*, i.e., $\mathcal{G}_{J \backslash \{j\}} = \texttt{BasisUpdate}(\mathcal{G}_{J \backslash \{j,\mathrm{s}[j]\}}, P_{\mathrm{s}[j]})$.

Proposition 3 and *Remark 4* can solve *Problem 5* if $L \neq \emptyset$, i.e., there exists $Q^{(l)} \in \mathcal{G}_J$ s.t. $(x - \alpha_j) \mid Q^{(l)}$. Therefore, it is necessary to determine whether $Q^{(l)} \in \mathcal{G}_J$ is divisible by $x - \alpha_j$. For this purpose, a simple criterion can be used.

Proposition 5 ([16]). Given $Q(x,y) = Q_{[0]}(x) + Q_{[1]}(x)y \in \mathcal{M}_J$, $(x - \alpha_j) \mid Q$ iff $Q_{[1]}(\alpha_j) = 0$.

If $L = \emptyset$, i.e., $Q^{(l)}_{[1]}(\alpha_j) \neq 0, \forall l$, then $\mathcal{G}_J$ can be transformed into another Gröbner basis of $\mathcal{M}_J$ with $L \neq \emptyset$. *Proposition 6* shows that a linear combination of the elements of $\mathcal{G}_J$ can generate a polynomial $\bar{Q} \in \mathcal{M}_J$ s.t. $\bar{Q}_{[1]}(\alpha_j) = 0$.

Proposition 6 ([16]). Given $\mathcal{G}_J = \{Q^{(0)}, Q^{(1)}\}$ and $j \in J$, let $\theta_j^{(l)} = Q^{(l)}_{[1]}(\alpha_j)$ for $l = 0, 1$. If $L = \{l \in \{0,1\} : \theta_j^{(l)} = 0\} = \emptyset$, let $\bar{Q} = Q^{(0)} - (\theta_j^{(0)}/\theta_j^{(1)})Q^{(1)}$. Then, $\bar{Q}_{[1]}(\alpha_j) = 0$.

Observe that $\mathrm{LM}_{2,3}\bar{Q} = \mathrm{LM}_{2,3}\mathcal{G}_J^{(+)}$. Hence, $\{\mathcal{G}_J^{(-)}, \bar{Q}\}$ forms a Gröbner basis of $\mathcal{M}_J$ with $L \neq \emptyset$ and it can produce $\mathcal{G}_{J \backslash \{j\}}$ by applying *Proposition 3*.

B. Fast SEP Construction

The fast SEP construction mainly consists of forward and backward computations. These computations generate the SEBPs that satisfy (4). Subsequently, the SEP is substantiated using both the SEBPs and the message based on (3).

1) Forward Computation: With the input of position set J, the forward computation yields the Gröbner basis $\mathcal{G}_J$. This can be achieved directly using *Algorithm 2*. Furthermore, if the

interpolation point set $\mathcal{P}_J$ has a semi-grid structure as defined below, the computation can be expedited.

Definition III *([15])*. Given a set of points $\mathcal{S} \subseteq \mathcal{P}$, let $\mathbb{A}(\mathcal{S}) = \{\alpha : (\alpha, \gamma) \in \mathcal{S}, \forall \gamma \in \mathbb{F}_q\}$ and $\mathbb{B}_\alpha(\mathcal{S}) = \{\beta : (\alpha, \beta) \in \mathcal{S}\}$ for $\alpha \in \mathbb{A}(\mathcal{S})$. Then, $\mathcal{S}$ is called a (maximal) semi-grid on $\mathcal{P}$, if $|\mathbb{B}_\alpha(\mathcal{S})| = 2, \forall \alpha \in \mathbb{A}(\mathcal{S})$.

Let $S_J = \{j \in J : \mathrm{s}[j] \neq j \text{ and } \mathrm{s}[j] \in J\}$. It can be seen that the point set $\mathcal{P}_{S_J}$ is a semi-grid on $\mathcal{P}_J$ and there is no other semi-grid on $\mathcal{P}_J$ containing $\mathcal{P}_{S_J}$, except for $\mathcal{P}_{S_J}$ itself. Choose certain representatives in S_J to form another position set S_J^*, s.t. $\mathbb{A}(\mathcal{P}_{S_J^*}) = \mathbb{A}(\mathcal{P}_{S_J})$ and $\mathrm{s}[j] \notin S_J^*, \forall j \in S_J^*$. It follows from *Proposition 7* that the elements of $\mathcal{G}_J$ have a common factor of $Q_{S_J}(x) = \prod_{j \in S_J^*}(x - \alpha_j)$.

Proposition 7. Given $\mathcal{G}_J = \{Q^{(l)} : l = 0, 1\}$ and $j \in J$, $(x - \alpha_j) \mid Q^{(l)}, \forall l$ iff $\{j, \mathrm{s}[j]\} \subseteq J$ and $j \neq \mathrm{s}[j]$.

Corollary 8. For any $Q^{(l)} \in \mathcal{G}_J$, $Q_{S_J} \mid Q^{(l)}$.

Based on $\mathcal{G}_J$, by recursively applying *Proposition 3* for every $j \in S_J^*$, $\mathcal{G}_{J \backslash S_J} = \{Q^{(l)}/Q_{S_J} : l = 0, 1\}$ can be obtained. Therefore, the computation of $\mathcal{G}_J$ can be divided into the computations of Q_{S_J} and $\mathcal{G}_{J \backslash S_J}$, respectively. The forward computation is summarized in *Algorithm 3*.

Algorithm 3: Forward Computation

Input: $\mathcal{P}, J$
Output: $\mathcal{G}_J$
Identify S_J and S_J^* from J;
$\mathcal{G}_{J \backslash S_J} = \{Q^{(0)}, Q^{(1)}\} \leftarrow$ **KötterInterp**$(\mathcal{P},\ J \backslash S_J)$;
$Q_{S_J} \leftarrow \prod_{j \in S_J^*}(x - \alpha_j)$;
$\mathcal{G}_J \leftarrow \{Q_{S_J} \cdot Q^{(0)}, Q_{S_J} \cdot Q^{(1)}\}$;

2) Backward Computation: With the input of Gröbner basis $\mathcal{G}_{J_k}$, the backward computation eliminates the constraint given by interpolation point P_j from $\mathcal{G}_{J_k}$, yielding $\mathcal{G}_{J_k \backslash \{j\}}$ for all $j \in J_k$. Subsequently, the SEBP $\mathcal{Q}_j$ is derived from $\mathcal{G}_{J_k \backslash \{j\}}$. Based on *Proposition 7*, the backward computation can be categorized into cases of whether $j \in S_{J_k}$ or not.

Case I: For $j \in S_{J_k}$, it holds that $\mathrm{s}[j] \in J_k$. Based on *Propositions 3* and *7*, $\mathcal{G}_{J_k \backslash \{j, \mathrm{s}[j]\}}$ can be derived from $\mathcal{G}_{J_k}$ as

$$\mathcal{G}_{J_k \backslash \{j, \mathrm{s}[j]\}} = \{\tilde{Q}^{(l)} = \frac{Q^{(l)}}{x - \alpha_j} : l = 0, 1\}. \tag{5}$$

Let $\Delta_{\mathrm{s}[j]}^{(l)} = \tilde{Q}^{(l)}(P_{\mathrm{s}[j]})$. As discussed in *Remark 4*, $\mathcal{G}_{J_k \backslash \{j\}} \neq \mathcal{G}_{J_k \backslash \{j, \mathrm{s}[j]\}}$ and there exists at least one $l \in \{0, 1\}$, s.t. $\Delta_{\mathrm{s}[j]}^{(l)} \neq 0$. Hence, $\mathcal{G}_{J_k \backslash \{j, \mathrm{s}[j]\}}$ can be updated to

$$\mathcal{G}_{J_k \backslash \{j\}} = \{\mathrm{F}_{\Delta_{\mathrm{s}[j]}^{(0)}, \Delta_{\mathrm{s}[j]}^{(1)}}(\tilde{Q}^{(0)}, \tilde{Q}^{(1)}), Q^{(l^*)}\} \tag{6}$$

using *Algorithm 1*, where $l^* = \arg\min_l \{\tilde{Q}^{(l)} : \Delta_{\mathrm{s}[j]}^{(l)} \neq 0\}$. Note that $Q^{(l^*)} \in \mathcal{G}_{J_k}$, and thus $Q^{(l^*)}(P_j) = 0$. According to constraint (4) in *Problem 3*, Q_j must satisfy $Q_j(P_j) \neq 0$. Therefore, $Q_j = \mathrm{F}_{\Delta_{\mathrm{s}[j]}^{(0)}, \Delta_{\mathrm{s}[j]}^{(1)}}(\tilde{Q}^{(0)}, \tilde{Q}^{(1)})$.

The formal derivative of $Q^{(l)}$ can be used to compute the evaluation $\Delta_j^{(l)} = \tilde{Q}^{(l)}(P_j)$ as

$$\tilde{Q}^{(l)}(P_j) = \frac{Q^{(l)}(\alpha_j, \beta_j)}{(x - \alpha_j)(\alpha_j, \beta_j)} = \frac{\partial Q^{(l)}}{\partial x}(P_j). \tag{7}$$

Furthermore, let $\Lambda_j = Q_j(P_j)$. It can be calculated by

$$Q_j(P_j) = \mathrm{F}_{\Delta_{\mathrm{s}[j]}^{(0)}, \Delta_{\mathrm{s}[j]}^{(1)}}(\Delta_j^{(0)}, \Delta_j^{(1)}). \tag{8}$$

Consequently, $\mathcal{Q}_j = Q_j/\Lambda_j$ can be determined.

Case II: For $j \in J_k \backslash S_{J_k}$, it holds that $\mathrm{s}[j] \notin J_k$. Based on *Propositions 5* and *7*, there exists at least one $l \in \{0, 1\}$, s.t. $\theta_j^{(l)} = Q_{[1]}^{(l)}(\alpha_j) \neq 0$. According to *Propositions 5* and *6*,

$$\mathrm{F}_{\theta_j^{(0)}, \theta_j^{(1)}}(Q^{(0)}, Q^{(1)}) = \begin{cases} Q^{(0)}, \text{ if } \theta_j^{(0)} = 0 \text{ and } \theta_j^{(1)} \neq 0 \\ Q^{(1)}, \text{ if } \theta_j^{(0)} \neq 0 \text{ and } \theta_j^{(1)} = 0 \\ \bar{Q}, \text{ if } \theta_j^{(0)} \neq 0 \text{ and } \theta_j^{(1)} \neq 0 \end{cases}$$

must be divisible by $x - \alpha$. Hence, based on *Proposition 3*,

$$\mathcal{G}_{J_k \backslash \{j\}} = \{\frac{\mathrm{F}_{\theta_j^{(0)}, \theta_j^{(1)}}(Q^{(0)}, Q^{(1)})}{x - \alpha_j}, Q^{(l')}\}, \tag{9}$$

where $l' = \arg\min_l \{Q^{(l)} : \theta_j^{(l)} \neq 0\}$. Similar to Case I, $Q_j = \mathrm{F}_{\theta_j^{(0)}, \theta_j^{(1)}}(Q^{(0)}, Q^{(1)})/(x - \alpha_j)$, and $\Lambda_j = Q_j(P_j)$ can be calculated by

$$Q_j(P_j) = \mathrm{F}_{\theta_j^{(0)}, \theta_j^{(1)}}(\frac{\partial Q^{(0)}}{\partial x}, \frac{\partial Q^{(1)}}{\partial x})(P_j). \tag{10}$$

Algorithm 4 summarizes the backward computation for SEP.

Algorithm 4: Backward Computation for SEP

Input: $\mathcal{P}, J_k, S_{J_k}, S_{J_k}^*, \mathcal{G}_{J_k}$
Output: $\{\mathcal{Q}_j : j \in J_k\}$
$Q_x^{(l)} \leftarrow \frac{\partial Q^{(l)}}{\partial x}$ **for** $l = 0, 1$;
for $j \in S_{J_k}^*$ **do**
  for $l = 0, 1$ **do**
    $\tilde{Q}^{(l)} \leftarrow Q^{(l)}/(x - \alpha_j)$;
    $\Delta^{(l)}(y) \leftarrow Q_x^{(l)}(\alpha_j, y)$;
    $\Delta_j^{(l)}, \Delta_{\mathrm{s}[j]}^{(l)} \leftarrow \Delta^{(l)}(\beta_j), \Delta^{(l)}(\beta_{\mathrm{s}[j]})$;
  for $j' = j, \mathrm{s}[j]$ **do**
    $\Lambda_{j'} \leftarrow \mathrm{F}_{\Delta_{\mathrm{s}[j']}^{(0)}, \Delta_{\mathrm{s}[j']}^{(1)}}(\Delta_{j'}^{(0)}, \Delta_{j'}^{(1)})$;
    $\mathcal{Q}_{j'} \leftarrow \mathrm{F}_{\Delta_{\mathrm{s}[j']}^{(0)}, \Delta_{\mathrm{s}[j']}^{(1)}}(\tilde{Q}^{(0)}, \tilde{Q}^{(1)})/\Lambda_{j'}$;
for $j \in J_k \backslash S_{J_k}$ **do**
  $\theta_j^{(l)} \leftarrow Q_{[1]}^{(l)}(\alpha_j)$ **for** $l = 0, 1$;
  $\Lambda_j \leftarrow \mathrm{F}_{\theta_j^{(0)}, \theta_j^{(1)}}(Q_x^{(0)}, Q_x^{(1)})(P_j)$;
  $\mathcal{Q}_j \leftarrow \mathrm{F}_{\theta_j^{(0)}, \theta_j^{(1)}}(Q^{(0)}, Q^{(1)})/(\Lambda_j(x - \alpha_j))$;

The following is an example demonstrating *Algorithm 4*.

Example 1. Given an elliptic curve $\mathcal{X} : Y^2 + Y = X^3$ defined over $\mathbb{F}_4$, its set of affine points $\mathcal{P}$ consists of $P_1 = (0, 0)$, $P_2 = (0, 1)$, $P_3 = (1, \sigma)$, $P_4 = (1, \sigma^2)$, $P_5 = (\sigma, \sigma)$, $P_6 = (\sigma, \sigma^2)$, $P_7 = (\sigma^2, \sigma)$ and $P_8 = (\sigma^2, \sigma^2)$, where σ is the primitive element of $\mathbb{F}_4$, satisfying $\sigma^2 = \sigma + 1$.

Given an information set $J_5 = \{1, 2, 3, 5, 8\}$ of elliptic code $\mathbb{C}_{\mathcal{L}}(\mathcal{P}, 5)$, the Gröbner basis $\mathcal{G}_{J_5}$ can be obtained using *Algorithm 3*. It consists of $Q^{(0)} = x + \sigma^2 x^2 + x^3 + \sigma xy$ and $Q^{(1)} = \sigma^2 x + x^2 + (\sigma x + \sigma^2 x^2)y$. The formal derivatives of $Q^{(0)}$ and $Q^{(1)}$ are $Q_x^{(0)} = 1 + x^2 + \sigma y$ and $Q_x^{(1)} = \sigma^2 + \sigma y$, respectively.

Since P_1 and P_2 have the same X-coordinate, $S_{J_5} = \{1, 2\}$. Based on (5), $\mathcal{G}_{J_5 \backslash \{1,2\}}$ can be determined. It consists of $\tilde{Q}^{(0)} = 1 + \sigma^2 x + x^2 + \sigma y$ and $\tilde{Q}^{(1)} = \sigma^2 + x + (\sigma + \sigma^2 x)y$. By evaluating $Q_x^{(0)}$ and $Q_x^{(1)}$ at P_1 and P_2, we have $\Delta_1^{(0)} = 1$, $\Delta_1^{(1)} = \sigma^2$, $\Delta_2^{(0)} = \sigma^2$ and $\Delta_2^{(1)} = 1$. According to (8), $\Lambda_1 = \sigma^2$ and $\Lambda_2 = 1$. With these results, the SEBPs $\mathcal{Q}_1$ and $\mathcal{Q}_2$ can be computed as follows:

$$\mathcal{Q}_1 = \frac{\tilde{Q}^{(0)} - (\Delta_2^{(0)}/\Delta_2^{(1)})\,\tilde{Q}^{(1)}}{\Lambda_1} = 1 + \sigma x^2 + (1 + \sigma^2 x)y,$$
$$\mathcal{Q}_2 = \frac{\tilde{Q}^{(0)} - (\Delta_1^{(0)}/\Delta_1^{(1)})\,\tilde{Q}^{(1)}}{\Lambda_2} = x + x^2 + (1 + x)y.$$

On the other hand, $Q_{[1]}^{(0)}(x) = \sigma x$ and $Q_{[1]}^{(1)}(x) = \sigma x + \sigma^2 x^2$. By evaluating $Q_{[1]}^{(0)}$ and $Q_{[1]}^{(1)}$ at P_3, P_5 and P_8, we have $\theta_3^{(0)} = \sigma$, $\theta_3^{(1)} = 1$, $\theta_5^{(0)} = \sigma^2$, $\theta_5^{(1)} = 1$, $\theta_8^{(0)} = 1$ and $\theta_8^{(1)} = 0$. According to (10), $\Lambda_3 = \sigma^2$, $\Lambda_5 = 1$ and $\Lambda_8 = \sigma$. Then, the SEBPs $\mathcal{Q}_3$, $\mathcal{Q}_5$ and $\mathcal{Q}_8$ can be computed as follows:

$$\mathcal{Q}_3 = \frac{Q^{(0)} - (\theta_3^{(0)}/\theta_3^{(1)})\,Q^{(1)}}{\Lambda_3(x - \alpha_3)} = \sigma x^2 + \sigma xy,$$
$$\mathcal{Q}_5 = \frac{Q^{(0)} - (\theta_5^{(0)}/\theta_5^{(1)})\,Q^{(1)}}{\Lambda_5(x - \alpha_5)} = \sigma x + x^2 + \sigma xy,$$
$$\mathcal{Q}_8 = \frac{Q^{(1)}}{\Lambda_8(x - \alpha_8)} = \sigma^2 x + \sigma xy.$$

C. Fast SGM Construction

The fast SGM construction can be derived from the fast SEP construction mentioned above. It consists of the forward computation given in *Algorithm 3* and a modified backward computation based on *Algorithm 4*. For the latter, it is necessary to store the common evaluation results.

Since the SGM $\mathbf{G}_{J_k}$ of $\mathbb{C}_{\mathcal{L}}(\mathcal{P}, k)$ with information set J_k consists of the evaluation images of the SEBPs, i.e., $\mathrm{E}(\mathcal{Q}_{j_1}), ..., \mathrm{E}(\mathcal{Q}_{j_k})$, the entry $g_{\mathrm{i}[j],i}$ of $\mathbf{G}_{J_k}$ is directly given by (4) for any $j, i \in J_k$. To compute the other entries efficiently, the following evaluation results need to be stored:

$$\mathcal{E} = \{E_i^{(l)} = Q^{(l)}(P_i) : l = 0, 1;\ i \in J_k^c\} \tag{11}$$

where $J_k^c = \{1, 2, ..., n\} \backslash J_k$.

Similar to the backward computation for SEP, the computation of $g_{\mathrm{i}[j],i}$ can be categorized into two cases.

Case I: For $j \in S_{J_k}$, $\Delta_{\mathrm{s}[j]}^{(l)}$ and Λ_j can be calculated by (7) and (8), respectively. Based on (5) and (6),

$$g_{\mathrm{i}[j],i} = \frac{\mathrm{F}_{\Delta_{\mathrm{s}[j]}^{(0)}, \Delta_{\mathrm{s}[j]}^{(1)}}(E_i^{(0)}, E_i^{(1)})}{\Lambda_j(\alpha_i - \alpha_j)}, \forall i \in J_k^c.$$

Case II: For $j \in J_k \backslash S_J$, $\theta_j^{(l)} = Q_{[1]}^{(l)}(\alpha_j)$, and Λ_j can be calculated by (10). Based on (9),

$$g_{\mathrm{i}[j],i} = \frac{\mathrm{F}_{\theta_j^{(0)}, \theta_j^{(1)}}(E_i^{(0)}, E_i^{(1)})}{\Lambda_j(\alpha_i - \alpha_j)}, \forall i \in J_k^c \backslash \{\mathrm{s}[j]\}.$$

If $\mathrm{s}[j] \neq j$, let $\Lambda_{\mathrm{s}[j]} = Q_j(P_{\mathrm{s}[j]})$, which can be calculated by replacing P_j in (10) with $P_{\mathrm{s}[j]}$. Then, $g_{\mathrm{i}[j],\mathrm{s}[j]} = \Lambda_{\mathrm{s}[j]}/\Lambda_j$.

It can be seen that the above computations can be performed in parallel, which is advantageous for hardware implementation.

V. Complexity Analysis

This section analyzes the $\mathbb{F}_q$-computational complexity of the proposed SEP and SGM construction algorithm.

1) Forward Computation (Algorithm 3): The complexity is no greater than computing $\mathcal{G}_{J_k}$ using *Algorithm 2*, which costs $\mathcal{O}(2 \cdot |J_k|^2) = \mathcal{O}(k^2)$ operations [21].

2) Backward Computation for SEP (Algorithm 4): For $j \in S_{J_k}$, computing $\tilde{Q}^{(l)}$ for $l = 0, 1$ costs $\mathcal{O}(2 \cdot \deg_{2,3} Q^{(l)})$ operations. Moreover, computing $\Delta_j^{(l)}$ and $\Delta_{\mathrm{s}[j]}^{(l)}$ for $l = 0, 1$ costs $\mathcal{O}(4 \cdot \deg_{2,3} Q_x^{(l)})$ operations. Note that $\deg_{2,3} Q_x^{(l)} < \deg_{2,3} Q^{(l)}$. With the above results, Λ_j can be calculated with $\mathcal{O}(1)$ operations. If both $\Delta_{\mathrm{s}[j]}^{(0)}$ and $\Delta_{\mathrm{s}[j]}^{(1)}$ are nonzero, then $Q_j = \tilde{Q}^{(0)} - \Delta_{\mathrm{s}[j]}^{(0)}/\Delta_{\mathrm{s}[j]}^{(1)} \cdot \tilde{Q}^{(1)}$, of which the computational complexity is

$$\mathcal{O}(\deg_{2,3} Q_j) = \mathcal{O}(\deg_{2,3} \mathcal{G}^{(+)}_{J_k \backslash \{j, \mathrm{s}[j]\}}) \subset \mathcal{O}(\deg_{2,3} \mathcal{G}^{(+)}_{J_k}).$$

Finally, computing $\mathcal{Q}_j = Q_j/\Lambda_j$ also costs $\mathcal{O}(\deg_{2,3} Q_j)$ operations. Based on *Lemma 2*, $\mathcal{O}(\deg_{2,3} Q^{(l)}) = \mathcal{O}(|J_k|) = \mathcal{O}(k)$ for $l = 0, 1$. Therefore, the complexity of the above computations is $\mathcal{O}(k)$.

For $j \in J_k \backslash S_{J_k}$, computing $\theta_j^{(l)}$ for $l = 0, 1$ costs $\mathcal{O}(2 \cdot \deg Q_{[1]}^{(l)}) \subset \mathcal{O}(\deg_{2,3} Q^{(l)}) = \mathcal{O}(k)$ operations. Furthermore, the computations of Λ_j, Q_j and $\mathcal{Q}_j$ are similar to the case of $j \in S_{J_k}$. Therefore, the complexity of these computations is also $\mathcal{O}(k)$.

Since there are k SEBPs $\mathcal{Q}_j$ required to be computed, the total complexity of backward computation for SEP is $\mathcal{O}(k^2)$.

3) Backward Computation for SGM: The algorithm starts with computing the common evaluation results (11), where each $E_i^{(l)}$ can be computed with $\mathcal{O}(\deg_{2,3} Q^{(l)}) = \mathcal{O}(k)$ operations. Since $|\mathcal{E}| = 2(n - k)$, the total complexity of computing (11) is $\mathcal{O}(k(n - k))$.

Furthermore, $\Delta_j^{(l)}$ (or $\theta_j^{(l)}$) and Λ_j need to be computed for all $j \in J_k$. Similar to the backward computation for SEP, these computations cost $k \cdot \mathcal{O}(k) = \mathcal{O}(k^2)$ operations. With the above results, each entry $g_{\mathrm{i}[j],i}$ of $\mathcal{G}_{J_k}$ can be calculated with $\mathcal{O}(1)$ operations, and thus computing the $k(n-k)$ entries costs $\mathcal{O}(k(n-k))$ operations. Consequently, the total complexity of backward computation for SGM is $\mathcal{O}(k^2) + \mathcal{O}(k(n - k)) = \mathcal{O}(kn)$.

Acknowledgement

This work is sponsored by the National Natural Science Foundation of China (NSFC) with project ID 62071498.

REFERENCES

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

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

[3] S. Sakata, J. Justesen, Y. Madelung and *et al.*, "Fast decoding of algebraic-geometric codes up to the designed minimum distance," *IEEE Trans. Inform. Theory*, vol. 41, no. 6, pp. 1672–1677, Sep. 1995.

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

[5] D. J. J. Versfeld, J. N. Ridley, H. C. Ferreira and *et al.*, "On systematic generator matrices for Reed–Solomon codes," *IEEE Trans. Inform. Theory*, vol. 56, no. 6, pp. 2549–2550, Jun. 2010.

[6] H. Matsui, "Unified system of encoding and decoding erasures and errors for algebraic geometry codes," in *Proc. Int. Symp. Inform. Theory and Its App. (ISITA)*, Taichung, Taiwan, Oct. 2010, pp. 1001–1006.

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

[8] Y. Wan, L. Chen, and F. Zhang, "Algebraic soft decoding of elliptic codes," *IEEE Trans. Commun.*, vol. 70, no. 3, pp. 1522–1534, Mar. 2022.

[9] 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. Inform. Theory*, vol. 56, no. 3, pp. 945–959, Mar. 2010.

[10] Y. Wan, L. Chen, and F. Zhang, "Algebraic Chase decoding of elliptic codes through computing the Gröbner Basis," in *Proc. IEEE Int. Symp. Inform. Theory (ISIT)*, Espoo, Finland, Jun. 2022, pp. 180–185.

[11] M. Fossorier and S. Lin, "Soft-decision decoding of linear block codes based on ordered statistics," *IEEE Trans. Inform. Theory*, vol. 41, no. 5, pp. 1379–1396, Sep. 1995.

[12] C. Heegard, J. Little, and K. Saints, "Systematic encoding via Gröbner bases for a class of algebraic-geometric Goppa codes," *IEEE Trans. Inform. Theory*, vol. 41, no. 6, pp. 1752–1761, Nov. 1995.

[13] H. Matsui and S. Mita, "Encoding via Gröbner bases and discrete Fourier transforms for several types of algebraic codes," in *Proc. IEEE Int. Symp. Inform. Theory (ISIT)*, Nice, France, Jun. 2007, pp. 2656–2660.

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

[15] P. Beelen, J. Rosenkilde, and G. Solomatov, "Fast Encoding of AG Codes Over C_{ab} Curves," *IEEE Trans. Inform. Theory*, vol. 67, no. 3, pp. 1641–1655, Mar. 2021.

[16] J. Zhu, X. Zhang, and Z. Wang, "Backward interpolation architecture for algebraic soft-decision Reed–Solomon decoding," *IEEE Trans. VLSI Systems*, vol. 17, no. 11, pp. 1602–1615, Nov. 2009.

[17] X. Zhang and Y. Zheng, "Generalized backward interpolation for algebraic soft-decision decoding of Reed-Solomon codes," *IEEE Trans. Commun.*, vol. 61, no. 1, pp. 13–23, Jan. 2013.

[18] L. C. Washington, *Elliptic Curves: Number Theory and Cryptography*, 2nd ed. Boca Raton, FL: Chapman & Hall/CRC, 2008.

[19] T. Høholdt, J. van Lint, and R. Pellikaan, "Algebraic geometry codes," in *Handbook of Coding Theory*, V. Pless, W. Huffman, and R. Brualdi, Eds. Amsterdam: Elsevier, 1998, pp. 871–961.

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

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