Fast Decoding of AG Codes

Beelen, Peter; Rosenkilde, Johan; Solomatov, Grigory

Published in:
IEEE Transactions on Information Theory

Link to article, DOI:
10.1109/TIT.2022.3188843

Publication date:
2022

Document Version
Peer reviewed version

This article has been accepted for publication in IEEE Transactions on Information Theory. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TIT.2022.3188843

Fast Decoding of AG Codes

Peter Beelen, Johan Rosenkilde, and Grigory Solomatov

***Abstract*—We present an efficient list decoding algorithm in the style of Guruswami-Sudan for algebraic geometry codes. Our decoder can decode any such code using $\tilde{\mathcal{O}}(s\ell^{\omega}\mu^{\omega-1}(n+g))$ operations in the underlying finite field, where n is the code length, g is the genus of the function field used to construct the code, s is the multiplicity parameter, ℓ is the designed list size and μ is the smallest positive element in the Weierstrass semigroup at some chosen place; the "soft-O" notation $\tilde{\mathcal{O}}(\cdot)$ is similar to the "big-O" notation $\mathcal{O}(\cdot)$, but ignores logarithmic factors. For the interpolation step, which constitutes the computational bottleneck of our approach, we use known algorithms for univariate polynomial matrices, while the root-finding step is solved using existing algorithms for root-finding over univariate power series.**

***Index Terms*—Algebraic Geometry Codes**

I. Introduction

Containing some of the best error-correcting codes currently known, algebraic geometry (AG) codes have received a lot of attention since their introduction by Goppa in [15]. The celebrated Guruswami-Sudan decoder [16] for these codes relies on an interpolation step as well as a root-finding step and is capable of decoding beyond half the designed minimum distance by returning a list of all codewords within a certain Hamming distance τ from the received word. In this article, we present an efficient realization of this decoder, achieving the best known complexity in the fully general setting of arbitrary AG codes. Moreover, except for the particularly simple case of Reed-Solomon codes, our decoder is at least as fast as all existing decoders which are tailored for specific families of codes. This article is based on a chapter of the PhD thesis of the third author [42].

Following the common practice, we will measure algorithmic complexity by asymptotically upper-bounding the number of arithmetic operations in the underlying finite field $\mathbb{F}_q$, relying on the *big-O* notation $\mathcal{O}(\cdot)$ as well as the *soft-O* notation $\tilde{\mathcal{O}}(\cdot)$, which ignores logarithmic factors. Formally, $\tilde{\mathcal{O}}(h) = \bigcup_{j=0}^{\infty} \mathcal{O}(h\log(h)^j)$ for any function $h : \mathbb{R}_{\geqslant 0} \to \mathbb{R}_{\geqslant 0}$, where $\mathbb{R}_{\geqslant 0}$ denotes the set of non-negative real numbers. Analogously to $\mathbb{R}_{\geqslant 0}$, we will also write $\mathbb{Z}_{\geqslant 0}$ and $\mathbb{Z}_{>0}$ for the non-negative and the positive integers respectively. Our complexity estimates will also involve ω, which denotes some real number such that the product of any two matrices in $\mathbb{F}_q^{m\times m}$ can be computed using $\mathcal{O}(m^{\omega})$ operations in $\mathbb{F}_q$. The naive algorithm for matrix multiplication yields $\omega = 3$, and it is clear that $\omega \geqslant 2$ in general; the current record with $\omega < 2.37286$ is due to [2].

Our decoder has the complexity $\tilde{\mathcal{O}}(\ell^{\omega+1}\mu^{\omega-1}(n+g))$, however, in a series of remarks throughout the article we explain how it can be slightly improved to $\tilde{\mathcal{O}}(s\ell^{\omega}\mu^{\omega-1}(n+g))$; here n is the code length, g is the genus of the function field used to construct the code, ℓ is the designed list size, $s \leqslant \ell$ is the multiplicity, and μ is the smallest element in the Weierstrass semigroup of some rational place P_∞ which is not one of the evaluation places. As we will see in Section II-B, the existence of such P_∞ can be assumed without any loss of generality.

A. Related work

As mentioned earlier, the paradigm of Guruswami-Sudan list decoding revolves around two main steps: interpolation and root-finding. As former is generally more computationally demanding, it has historically received the most attention. Several authors, including [1], [10], [27], [33], [34], formulated the interpolation step as a problem of finding a polynomial, minimal with respect to a weighted monomial order, in a certain vanishing ideal. Prompted by this, Lee and O'Sullivan developed a technique for obtaining such a polynomial from a Gröbner basis (of $\mathbb{F}_q[x]$-modules), that was itself computed starting from a particular generating set – first for RS codes [25], and then for one-point Hermitian codes [26]. The complexity of this strategy was further improved by Beelen and Brander in [4] by utilizing Alekhnovich's algorithm for row reduction of polynomial matrices [1]. Furthermore, their decoder was applicable to the wider family of one-point codes over C_{ab} curves, making it more general. Specializing back to one-point Hermitian codes, Rosenkilde and Beelen [32] sped up this approach even more by delegating the row-reduction phase to the algorithm by Giorgi, Jeannerod and Villard [13], which is more efficient than the one by Alekhnovich. Doing this required additional improvements to keep up with the new target complexity, including efficient computation of the initial $\mathbb{F}_q[x]$-basis, as well as a way of handling fractional weights. The result was the first list-decoder of one-point Hermitian codes having sub-quadratic complexity in the code length. In the current article, we generalize the tools from [32] to be applicable to all AG codes, relying on the conceptual framework from [24] to represent function field elements using *Apéry systems*.

Before shifting our attention to the root-finding step, we ought to mention the multivariate interpolation algorithm by

Peter Beelen is with the Department of Applied Mathematics and Computer Science, Technical University of Denmark, 2800 Kongens Lyngby, Denmark (e-mail: pabe@dtu.dk).

Johan Rosenkilde was with the Department of Applied Mathematics and Computer Science, Technical University of Denmark (DTU), 2800 Kongens Lyngby, Denmark. He is now with GitHub Inc., San Francisco, CA 94107 USA (email: jsrn@jsrn.dk).

Grigory Solomatov was with the Department of Applied Mathematics and Computer Science, Technical University of Denmark (DTU), 2800 Kongens Lyngby, Denmark. He is now with Department Electrical Engineering - Systems, Tel Aviv University, 6997801 Tel Aviv, Israel (email: grigory93@gmail.com).

The authors would like to acknowledge the support from The Danish Council for Independent Research (DFF-FNU) for the project *Correcting on a Curve*, Grant No. 8021-00030B.

Chowdhury, Jeannerod, Neiger, Schost and Villard [9] – it was the first to enable the currently best complexity in the special case of RS codes, albeit in a probabilistic manner. A deterministic algorithm with the same complexity was later given in [21].

Some of the earliest root-finding algorithms for Guruswami-Sudan list-decoding include Roth and Ruckenstein's [36] as well as Gao and Shokrollahi's [12]. Alekhnovich described in [1] an efficient approach for computing the $\mathbb{F}_q[\![x]\!]$-roots modulo x^β of a polynomial $Q \in \mathbb{F}_q[\![x]\!][z]$; its complexity was shown in [32] to be $\tilde{\mathcal{O}}(\beta^2\ell)$ operations in $\mathbb{F}_q$, where ℓ is the z-degree of Q. Another technique by Berthomieu, Lecerf and Quintin [7] achieved the cost $\tilde{\mathcal{O}}(\beta\ell^2)$. In this article, we rely on the algorithm by Neiger, Rosenkilde and Schost [29], whose complexity of $\tilde{\mathcal{O}}(\beta\ell)$ operations is provably quasi-optimal.

The complexity of our decoder is at least as good as, and often faster than, the complexity of previous decoders based on the Guruswami-Sudan paradigm. As far as we know, there is only one exception: in the case of RS codes, the complexity of the algorithms from [9] or [21] is a factor of ℓ/s better. To illustrate the strength and versatility of our results, in Section VI, examples are given of the list decoding of various families of AG codes. One further remark should be made, namely the case of bounded distance decoding. Setting $s = \ell = 1$ and assuming that $g \in \mathcal{O}(n)$, the complexity our decoder simplifies to $\tilde{\mathcal{O}}(\mu^{\omega-1}n)$. In this case, the decoder can always correct up to $(d^* - 1 - g)/2$ errors, where d^* denotes the designed minimum distance of an AG code, also known as the Goppa bound. The same decoding radius is achieved in [38] with complexity $\mathcal{O}(\mu n^2)$. Since $\mu \leqslant g$ and we assumed $g \in \mathcal{O}(n)$, our complexity is better. However, Sakata's extension of the Berlekamp-Massey decoder [37], [39] yields a decoding algorithm able to correct up to at least $(d^* - 1)/2$ errors. In [18], the complexity in the case of certain so-called one-point AG codes is $\mathcal{O}(\mu n^2 + q^{t+1}(a_1 + \cdots + a_t) + tnq^t)$, where $a_1, \ldots, a_t$ form a minimal set of generators of the Weierstrass semigroup at P_∞. To achieve the same decoding radius with our decoder, we could choose s and ℓ in $\mathcal{O}(g)$, but doing so might not be as efficient, since our complexity would then increase by a factor of $\mathcal{O}(g^{\omega+1}) \subseteq \mathcal{O}(n^{\omega+1})$.

B. Strategy outline and contributions

With the aim of making the exposition easier in the subsequent sections of the article, we now present an overview of the main steps in the proposed decoder. This consists of a way of simplifying the general setting as well as a way of efficiently carrying out the classical steps of interpolation and root-finding. The complete decoder is presented in Section VI, where it is also exemplified for special cases of AG codes.

- **Simplified setting:** In Section II-B, we show how extending the constant field $\mathbb{F}_q$ allows us to make certain simplifying assumptions. The important takeaway here is that no generality is sacrificed in the process, while only a minor penalty is introduced into the computational complexity. In return, we may assume existence of certain rational places, as well as existence of a special function field element x with controlled zeroes and poles. Having access to additional rational places is useful for a variety of reasons, among which is efficient multiplication of function field elements in a pointwise manner; the carefully chosen function x acts as a fundamental building block in the way we represent function field elements.
- **Interpolation step:** This is the most involved part of the article and requires all of the computational tools from Section V – except for Section V-F, which deals with root-finding. In Section IV-A, it is explained how the interpolation step can be viewed as a problem of finding a "small" element Q in a certain interpolation module whose underlying ring consists of all functions that have no poles except for possibly at a fixed rational place P_∞. This ring, denoted by Я, is itself a free module over $\mathbb{F}_q[x]$, which essentially means that we can represent everything as tuples of univariate polynomials. The computational path for obtaining Q boils down to first constructing a generating set of the interpolation module over Я, then expanding this to a generating set over $\mathbb{F}_q[x]$, and finally, using efficient algorithms for matrices over $\mathbb{F}_q[x]$ to reduce this generating set to a "small" basis that contains a satisfactory Q.
- **Root-finding step:** Structurally, the obtained Q is a univariate polynomial whose coefficients are function field elements; and according to the Guruswami-Sudan paradigm, list decoding reduces to finding the roots of this polynomial. We accomplish this by expressing the coefficients of Q as power series in x, which is always possible in our simplified setting in which x is a local parameter of some appropriate rational place. An existing algorithm for root-finding over the ring of power-series is then used to obtain the sought roots, albeit represented as power series; the final step of our decoder therefore consists of converting these roots back to the original representation as well as discarding any potential "spurious" solutions. All of this is detailed in Section V-F.

Our decoder relies on a mixture of new and existing results; the novel contributions include:

- reduction of the fully general setting to a simpler one (Section II-B),
- an algorithm for encoding general AG codes with complexity $\tilde{\mathcal{O}}(\mu n)$ (Section V-A),
- an interpolation algorithm with complexity $\tilde{\mathcal{O}}(\mu^{\omega-1}(n + g))$ (Section V-B),
- a root-finding algorithm with complexity $\tilde{\mathcal{O}}(\ell^2\mu^{\omega-1}(n + g))$ (Section V-F),
- an algorithm for computing an $\mathbb{F}_q[x]$-basis of $\langle h \rangle_{\text{Я}}$ for any function h (Section V-D).

Not counting precomputation, all of the algorithms above are sufficiently efficient to reach our target cost. Although the cost of precomputation has not been investigated in detail, it is not expected to be much more expensive than that of Gaussian elimination. A list of all precomputed objects can be found in Section VI.

II. Preliminaries

A. AG codes

Let $\mathbb{F}_q$ be a finite field with q elements, where q is a power of a prime number p. Further, let F be a function field of genus g and full constant field $\mathbb{F}_q$. As is common, we denote by $\mathbb{P}_F$ the set of places of F.

For any divisor $A = \sum_i n_i Q_i$ of F, we denote by $\mathrm{supp}(A)$ the support of A, which consists of all places Q_i such that $n_i \neq 0$. A divisor A is called effective, denoted by $A \geqslant 0$, if for all i it holds $n_i \geqslant 0$. Further, the degree of A, is defined as $\deg(A) = \sum_i n_i \deg(Q_i)$, where $\deg(Q_i)$ denotes the degree of the place Q_i.

The well-known Riemann-Roch space of a divisor A is given by

$$\mathcal{L}(A) = \{f \in F \backslash \{0\} \mid (f) + A \geqslant 0\} \cup \{0\} \ ,$$

where (f) denotes the divisor of f. The Riemann-Roch space $\mathcal{L}(A)$ is a vector space over $\mathbb{F}_q$, whose dimension will be denoted by $l(A)$. The theorem of Riemann-Roch [43, Theorem 1.5.15] implies that $l(A) \geqslant \deg(A) + 1 - g$ and that equality holds if $\deg(A) \geqslant 2g - 1$. Moreover $l(A) = 0$ if $\deg(A) < 0$ since the degree of a principal divisor is zero.

Definition II.1. *Assume that F has at least n rational places, say $P_1, \ldots, P_n$ and write $D = P_1 + \cdots + P_n$. Further, let G be a divisor of F such that $\mathrm{supp}(G) \cap \mathrm{supp}(D) = \varnothing$[1]. Then we define*

$$\mathcal{C}_{\mathcal{L}}(D, G) = \{\mathrm{ev}_D(f) \mid f \in \mathcal{L}(G)\} \subset \mathbb{F}_q^n \ ,$$

where for any $f \in \mathcal{L}(G)$, $\mathrm{ev}_D(f) = (f(P_1), \ldots, f(P_n)) \in \mathbb{F}_q^n$.

For future reference, we state some properties of this code, see [43, Chapter 2] for details. First of all, it is well-known that this code has minimum distance at least $n - \deg(G)$. Since the kernel of ev_D is $\mathcal{L}(G - D)$, the dimension of the code equals $l(G) - l(G - D)$. In particular, $\mathcal{C}_{\mathcal{L}}(D, G)$ is the zero code if $\deg(G) < 0$. Further, using the theorem of Riemann-Roch, we see that $\dim(\mathcal{C}_{\mathcal{L}}(D, G)) = n$, i.e. $\mathcal{C}_{\mathcal{L}}(D, G) = \mathbb{F}_q^n$, whenever $\deg(G) \geqslant n + 2g - 1$. Because of this, we may assume

$$0 \leqslant \deg(G) \leqslant n + 2g - 1. \tag{II.1}$$

Remark II.2. *In his original construction, Goppa considered AG codes $\mathcal{C}_{\Omega}(D, G)$ defined using residues of certain differentials. These codes can also be obtained as evaluation codes [43, Proposition 2.2.10]. Hence our decoder can also handle codes of the form $\mathcal{C}_{\Omega}(D, G)$.*

B. Reduction to a simpler setting

In this subsection, we will show that without significant increase of decoding complexity, we can assume several things about the function field F and the AG code $\mathcal{C}_{\mathcal{L}}(D, G)$ that will make the exposition of our decoding algorithm simpler later on. For example, it will be convenient to have an additional rational place P_∞ of F that is not used in the evaluation map ev_D. In fact, for some of our later algorithms, it will be convenient to have additional rational places as well. An easy way out is to increase the constant field $\mathbb{F}_q$ to $\mathbb{F}_{q^e}$ for some small value of e, thus introducing new rational places that can be used as additional rational places. We will denote by $F\mathbb{F}_{q^e}$, the function field obtained from F by extending the constant field to $\mathbb{F}_{q^e}$.

As far as decoding is concerned, the AG code $\mathcal{C}_{\mathcal{L}}(D, G)$ is in a trivial way a subcode (not $\mathbb{F}_{q^e}$-linear, but $\mathbb{F}_q$-linear) of the AG code obtained from the function field $F\mathbb{F}_{q^e}$ using the divisors $\mathrm{Con}(D)$ and $\mathrm{Con}(G)$, where Con denotes the conorm with respect to $F\mathbb{F}_{q^e}/F$, [43, Definition 3.1.8]. Since all places in $\mathrm{supp}(D)$ are rational, we may with slight abuse of notation write $\mathrm{Con}(D) = D$. Hence if for a given τ, one has a list decoding algorithm for $\mathcal{C}_{\mathcal{L}}(D, \mathrm{Con}(G))$ that produces all codewords at distance at most τ from a received word, one immediately obtains a list decoding algorithm for $\mathcal{C}_{\mathcal{L}}(D, G)$. However, since multiplication of two elements in $\mathbb{F}_{q^e}$ can be done in $\tilde{\mathcal{O}}(e)$ operations in $\mathbb{F}_q$ [8], the value of e should be small for complexity reasons. Therefore we now give a series of lemmas, each aiming to show that for small e, simplifying assumptions can be made about the function field F and the code $\mathcal{C}_{\mathcal{L}}(D, G)$.

Lemma II.3. *Let F be a function field over $\mathbb{F}_q$ of genus g and denote by N_e the number of rational places of the function field $F\mathbb{F}_{q^e}$ over $\mathbb{F}_{q^e}$. If $N, e \in \mathbb{Z}_{>0}$ are such that $e \geqslant 2\log_q \max\{N, 2g+1\}$, then $N_e > N$.*

Proof. The Hasse-Weil bound $|(q^e+1) - N_e| \leqslant 2q^{e/2}g$ implies that

$$\begin{aligned} \log_q N_e &> \log_q(q^e - 2q^{e/2}g) \\ &= e/2 + \log_q(q^{e/2} - 2g) \\ &\geqslant e/2 \geqslant \log_q N \ . \end{aligned}$$

□

Now we show that if the function field F has sufficiently many rational places, then one of them has particularly simple local parameter. Recall that a function $f \in F$ is called a local parameter for a place P if $v_P(f) = 1$, where v_P denotes the valuation at P.

Lemma II.4. *Let F be a function field over $\mathbb{F}_q$ of genus g having a rational place P_∞. Let μ be the smallest positive element from the Weierstrass semigroup of P_∞. Any set containing at least $3g+1$ rational places distinct from P_∞, contains a place P_0 with local parameter from $\mathcal{L}(\mu P_\infty)$.*

Proof. Let $x \in \mathcal{L}(\mu P_\infty)$ be a function satisfying $v_{P_\infty}(x) = -\mu$. First of all, note that the extension $F/\mathbb{F}_q(x)$ is separable. Indeed, assume that $F/\mathbb{F}_q(x)$ is inseparable. Then by the general theory of inseparable extensions, we can find an intermediate field E such that $E/\mathbb{F}_q(x)$ is separable and F/E is purely inseparable. Then by [43, Proposition 3.10.2], $\mathbb{F}_q(x) \subseteq F^p = \{f^p \mid f \in F\}$, where p is the characteristic. Hence $x = y^p$ for some $y \in F$. Since x has a pole at P_∞ only of order μ, this would imply that the function y also has a pole at P_∞ only of order μ/p. This gives a contradiction with the minimality of μ.

[1]The assumption that $\mathrm{supp}(G) \cap \mathrm{supp}(D) = \varnothing$ can be removed [43, Remark 2.2.15], but doing so does not give new AG codes up to monomial equivalence.

The above implies that the Hurwitz genus formula (see for example [43, Corollary 3.4.14]) applies to the extension $F/\mathbb{F}_q(x)$. To prove the lemma, we estimate the genus of F using this formula. We write $Q_\infty = P_\infty \cap \mathbb{F}_q(x)$, which is the place at infinity of the rational function field $\mathbb{F}_q(x)$. Since $v_{P_\infty}(x) = -\mu$ and $v_{Q_\infty}(x) = -1$, we see that $e(P_\infty|Q_\infty) = \mu$. Now suppose we have $N > 3g$ rational places distinct from P_∞, say $P_1, \ldots, P_N$. We write $Q_i = P_i \cap \mathbb{F}_q(x)$ for their restrictions to $\mathbb{F}_q(x)$. For these rational places, we have

$$v_{P_i}(x - x(P_i)) = e(P_i|Q_i)v_{Q_i}(x - x(P_i)) = e(P_i|Q_i).$$

Suppose that for every rational place P_i it holds that $e(P_i|Q_i) \geqslant 2$.

Since $\mu = [F : \mathbb{F}_q(x)]$ by [43, Theorem 1.4.11], the Hurwitz genus formula combined with the estimate $d(P_i|Q_i) \geqslant e(P_i|Q_i) - 1$ implies that

$$\begin{aligned} 2g - 2 &\geqslant -2[F : \mathbb{F}_q(x)] + d(P_\infty|Q_\infty) + \sum_{i=1}^{N} d(P_i|Q_i) \\ &\geqslant -2\mu + (\mu - 1) + N(2 - 1)\ . \end{aligned}$$

Since $\mu \leqslant g + 1$, we conclude that $N \leqslant 3g$, a contradiction. Hence for one of the places $P_1, \ldots, P_N$ we have $v_{P_i}(x - x(P_i)) = 1$. □

To motivate Lemma II.4, recall from Lemma II.3 that by extending our base field $\mathbb{F}_q$ to $\mathbb{F}_{q^e}$, we can easily "create" as many new rational places as we need without compromising our target complexity. By doing this, we can ensure that there exists a function $x \in \mathcal{L}(\mu P_\infty)$ which is also a local parameter of some rational place P_0 not in $\operatorname{supp}(G)$. As we will see in Section V, membership of $x \in \mathcal{L}(\mu P_\infty)$ allows us to impose an $\mathbb{F}_q[x]$-module structure on the interpolation step of Guruswami-Sudan decoding. In Section V we will use the assumption that x is a local parameter of P_0 to represent certain functions in F as power series in $\mathbb{F}_q[\![x]\!]$, which allows us to solve the root-finding step efficiently.

Next we consider a lemma showing that we can assume that the divisor G used to define the AG code $\mathcal{C}_\mathcal{L}(D, G)$ is effective unless the code is degenerate. We call a code $\mathcal{C}$ degenerate if there exists i such that $c_i = 0$ for any codeword $c = (c_1, \ldots, c_n) \in \mathcal{C}$. In particular the trivial code containing only the zero codeword is degenerate.

Lemma II.5. *Let the function field F and divisors G and D be as before. Then either, the AG code $\mathcal{C}_\mathcal{L}(D, G)$ is degenerate or $\mathcal{C}_\mathcal{L}(D, \operatorname{Con}(G))$ is monomially equivalent over $\mathbb{F}_{q^e}$ with $e \geqslant 1 + \lceil \log_q(n) \rceil$, to an AG code $\mathcal{C}_\mathcal{L}(D, G')$, where G' is an effective divisor of $F\mathbb{F}_{q^e}$ of degree $\deg(G)$.*

Proof. Consider the finite field extension $\mathbb{F}_{q^e}/\mathbb{F}_q$ and for convenience, let us write $\mathcal{C} = \mathcal{C}_\mathcal{L}(D, \operatorname{Con}(G))$ as well as $\mathcal{C}_i = \{c \in \mathcal{C} \mid c_i = 0\}$. If $\mathcal{C}_\mathcal{L}(D, G)$ is nondegenerate, then so is $\mathcal{C}$. In this case $\mathcal{C} \neq \mathcal{C}_i$ for all i. If every codeword in $\mathcal{C}$ has at least one zero coordinate, then $\mathcal{C} = \bigcup_{i=1}^n \mathcal{C}_i$, which implies that $(q^e)^k \leqslant n(q^e)^{k-1}$ with $k = \dim \mathcal{C}$. We see that in this case $q^e \leqslant n$, implying $e \leqslant \log_q(n)$. This contradiction shows that $\mathcal{C}$ contains a codeword of full Hamming weight n, say $c = \operatorname{ev}_D(\tilde{f})$ for some $\tilde{f} \in \mathcal{L}(\operatorname{Con}(G))$. Since by construction $\tilde{f}(P_i) \neq 0$ for all i, we see that the codes $\mathcal{C}$ and $\mathcal{C}_\mathcal{L}(D, \operatorname{Con}(G) + (\tilde{f}))$ are monomially equivalent using the map $(c_1, \ldots, c_n) \mapsto (\tilde{f}(P_1)c_1, \ldots, \tilde{f}(P_n)c_n)$. Note that the divisor $G' = \operatorname{Con}(G) + (\tilde{f})$ is effective and has support disjoint from D. □

Degenerate codes are not very interesting from the error-correcting point of view. Indeed, if the i-th coordinate of all codewords is zero, it is trivial to correct errors in that position. On the other hand that position does not carry any information, so one might as well consider the punctured code where such a position has been removed, which will have the same dimension and minimum distance. Decoding a degenerate code can therefore be reduced using puncturing to decoding a nondegenerate code. Note that since the codes $\mathcal{C}_\mathcal{L}(D, \operatorname{Con}(G))$ and $\mathcal{C}_\mathcal{L}(D, G')$ are monomially equivalent, any (list) decoding algorithm for $\mathcal{C}_\mathcal{L}(D, G')$ immediately gives a (list) decoding algorithm for $\mathcal{C}_\mathcal{L}(D, \operatorname{Con}(G))$. The added complexity is that of dividing and multiplying with the column multipliers $\tilde{f}(P_i)$, which only costs $\mathcal{O}(n)$ operations in $\mathbb{F}_{q^e}$ and hence $\tilde{\mathcal{O}}(ne)$ operations in $\mathbb{F}_q$. Moreover, as we will see, we will be able to choose e so small that it will not affect the decoding complexity at all in the $\tilde{\mathcal{O}}$ notation.

Now we state the simplifying assumptions and notation that will be used in the remainder of this article.

1) We assume that G is an effective divisor, whose degree satisfies equation (II.1).
2) We assume that apart from the rational places in $D = P_1 + \cdots + P_n$, the function field F has at least one more rational place P_∞. The place P_∞ may or may not be in $\operatorname{supp}(G)$.
3) There exists a rational place P_0 of F which has x as a local parameter, where $x \in F$ is a function with pole at P_∞ only of minimal pole order μ. The place P_0 may be in $\operatorname{supp}(D)$, but is not in $\operatorname{supp}(G)$.

Let us quickly assess the size of the needed extension degree e in order to satisfy all three item simultaneously. Although one likely can do better, for our purposes it is sufficient to pick $e = e_1e_2e_3$, where e_1, e_2, e_3 are given below: to satisfy the first item, we extend $\mathbb{F}_q$ to $\mathbb{F}_{q^{e_1}}$, where $e_1 = 1 + \lceil \log_q(n) \rceil$, using Lemma II.5. To satisfy the second item, we apply Lemma II.3 with $N = n + 1$. Hence we can choose $e_2 = \lceil 2\log_{q^{e_1}} \max\{n + 1, 2g + 1\} \rceil$ and extend $\mathbb{F}_{q^{e_1}}$ to $\mathbb{F}_{q^{e_1e_2}}$. For the third item, we need apart from P_∞ and possible rational places in $\operatorname{supp}(G)$, an additional $3g + 1$ rational places. Since G is effective, we can apply Lemma II.3 with $N = 1 + \deg(G) + 3g + 1$, so using equation (II.1), we can choose $e_3 = \lceil 2\log_{q^{e_1e_2}}(5g + 1 + n) \rceil$ extending $\mathbb{F}_{q^{e_1e_2}}$ to $\mathbb{F}_{q^e}$. Using that $\log_{q^f}(A) = \log_q(A)/f$, it is easy to see that

$$\begin{aligned} e &= e_1e_2e_3 \\ &\leqslant 2\log_q(5g + 1 + n) + e_1e_2 \\ &\leqslant 2\log_q(5g + 1 + n) + 2\log_q \max\{n + 1, 2g + 1\} + e_1 \\ &\leqslant 2\log_q(5g + 1 + n) + 2\log_q \max\{n + 1, 2g + 1\} \\ &\qquad + \log_q(n) + 2. \end{aligned}$$

Hence the overall conclusion is that in terms of complexity only a logarithmic factor in $n+g$ is introduced when reducing from the general case to the simpler setting. In the remainder of this article, instead of writing $\mathbb{F}_{q^e}$, we will simply write $\mathbb{F}_q$ and assume q is large enough so that all three simplifying assumptions stated above are satisfied.

C. Shifted Popov forms of polynomial matrices

Our decoder relies on efficient algorithms for (free) $\mathbb{F}_q[x]$-submodules of $\mathbb{F}_q[x]^m$; in the current subsection, we present well known results and definitions that we need needed for our use cases. For a comprehensive introduction, the reader is referred to [44] and the references within.

We begin with a definition which, among other things, allows us to measure "size" of elements in $\mathbb{F}_q[x]^m$.

Definition II.6. *For any polynomial vector $\boldsymbol{v} = (v_1, \ldots, v_m) \in \mathbb{F}_q[x]^m$ and any $\boldsymbol{s} = (s_1, \ldots, s_m) \in \mathbb{Z}^m$ (which we refer to as a* shift*), we define the $\boldsymbol{s}$-*degree *of $\boldsymbol{v}$ as*

$$\deg_{\boldsymbol{s}} \boldsymbol{v} = \max_k \{\deg v_k + s_k\} \ .$$

*Furthermore, if $k \in \{1, \ldots, m\}$ is maximal such that $\deg v_k + s_k = \deg_{\boldsymbol{s}} \boldsymbol{v}$, then we say that v_k is the $\boldsymbol{s}$-*pivot *of $\boldsymbol{v}$, and k is its $\boldsymbol{s}$-*pivot index*. If $\boldsymbol{s} = 0$, then we might omit writing $\boldsymbol{s}$ in the above notation, i.e. we might simply write:* pivot, pivot index *and* degree*, denoting the latter by $\deg \boldsymbol{v} := \deg_{\boldsymbol{0}} \boldsymbol{v}$.*

Any $\mathbb{F}_q[x]$-basis of a submodule $\mathcal{V} \subseteq \mathbb{F}_q[x]^m$ of rank m can be described using a nonsingular polynomial matrix $\boldsymbol{V} \in \mathbb{F}_q[x]^{m\times m}$ by identifying the basis elements with the rows of $\boldsymbol{V}$. This way, $\mathcal{V}$ is viewed as the $\mathbb{F}_q[x]$-row space of $\boldsymbol{V}$. We will be interested in obtaining the basis whose elements are "smallest possible"; the following definition makes this notion precise in the context of polynomial matrices.

Definition II.7. *Given a shift $\boldsymbol{s} \in \mathbb{Z}^m$, a nonsingular matrix $\boldsymbol{P} \in \mathbb{F}_q[x]^{m\times m}$ is said to be in $\boldsymbol{s}$-*Popov form *if all of the $\boldsymbol{s}$-pivots of its rows lie on the diagonal, are monic and have degrees strictly greater than all other entries in their respective columns. Furthermore, if $\boldsymbol{P}$ shares its $\mathbb{F}_q[x]$-row space with some matrix $\boldsymbol{V} \in \mathbb{F}_q[x]^{r\times m}$, where $m \leqslant r$, then $\boldsymbol{P}$ is said to be* the $\boldsymbol{s}$-Popov form *of $\boldsymbol{V}$.*

Below, we summarize a few important structural properties of shifted Popov forms.

Proposition II.8 ([44, Section 1.1])**.** *For any nonsingular matrix $\boldsymbol{V} \in \mathbb{F}_q[x]^{m\times m}$ and any shift $\boldsymbol{s} \in \mathbb{Z}^m$, there exists a unique matrix $\boldsymbol{P} \in \mathbb{F}_q[x]^{m\times m}$ in $\boldsymbol{s}$-Popov form having the same $\mathbb{F}_q[x]$-row space as $\boldsymbol{V}$. Furthermore, $\boldsymbol{P}$ has minimal shifted row degrees in the following sense: for any $\boldsymbol{V} \in \mathbb{F}_q[x]^{m\times m}$ with the same row space as $\boldsymbol{P}$, there exists a bijection between the rows of the two matrices such that the $\boldsymbol{s}$-degree of any row of $\boldsymbol{V}$ is no smaller than that of the corresponding row of $\boldsymbol{P}$. Finally, for any nonzero vector $\boldsymbol{v} \in \mathbb{F}_q[x]^{1\times m}$ in the row space of $\boldsymbol{P}$ with $\boldsymbol{s}$-pivot index k it holds that $\deg_{\boldsymbol{s}} \boldsymbol{v} \geqslant \deg_{\boldsymbol{s}} \boldsymbol{p}^{(k)}$, where $\boldsymbol{p}^{(k)}$ denotes the k-th row of $\boldsymbol{P}$.*

We conclude this subsection with a few complexity bounds.

Proposition II.9 ([30, Theorem 1.3])**.** *There is a deterministic algorithm which for any shift $\boldsymbol{s} \in \mathbb{Z}^m$ computes the $\boldsymbol{s}$-Popov form of any nonsingular matrix $\boldsymbol{V} \in \mathbb{F}_q[x]^{m\times m}$ using $\tilde{\mathcal{O}}(m^\omega \deg \boldsymbol{V})$ operations in $\mathbb{F}_q$, where $\deg \boldsymbol{V}$ denotes the maximal degree among all entries in $\boldsymbol{V}$.*

Proposition II.10 ([46])**.** *There is a deterministic algorithm which for any matrix $\boldsymbol{V} \in \mathbb{F}_q[x]^{r\times m}$ with $m \leqslant r$ computes an $\mathbb{F}_q[x]$-basis of the row space of $\boldsymbol{V}$ using $\tilde{\mathcal{O}}(rm^{\omega-1} \deg \boldsymbol{V})$ operations in $\mathbb{F}_q$.*

Combining Proposition II.10 with Proposition II.9, we obtain the following:

Corollary II.11. *For any shift $\boldsymbol{s} \in \mathbb{Z}^m$ and any matrix $\boldsymbol{V} \in \mathbb{F}_q[x]^{r\times m}$ with rank $m \leqslant r$, we can compute the $\boldsymbol{s}$-Popov form of $\boldsymbol{V}$ using $\tilde{\mathcal{O}}(rm^{\omega-1} \deg \boldsymbol{V})$ operations in $\mathbb{F}_q$.*

III. Representation of function field elements

For any divisor A of F, let $\text{Я}(A) = \bigcup_{m=-\infty}^{\infty} \mathcal{L}(mP_\infty + A)$ and let $\text{Я} = \text{Я}(0)$. Note that Я is a ring and $\text{Я}(A)$ a Я-module. In fact more can be said: Я is a Dedekind domain and $\text{Я}(A)$ is a fractional ideal of Я, [31, Section 1.2].

Modules of the form $\text{Я}(A)$ are essentially already considered for decoding in [23], also see [5], [24]. As in [24], for any nonzero $a \in \text{Я}(A)$ we denote by $\delta_A(a)$ the smallest integer m such that $a \in \mathcal{L}(mP_\infty + A)$, i.e. $\delta_A(a) = -v_{P_\infty}(a) - v_{P_\infty}(A)$ and let $\delta(a) = \delta_0(a) = -v_{P_\infty}(a)$. We will take as convention that $\delta_A(0) = -\infty$. Note that for any $a \in \text{Я}(A)$ and $b \in \text{Я}(B)$, one has $\delta_{A+B}(ab) = \delta_A(a) + \delta_B(b)$.

It is well known that any fractional ideal of a Dedekind domain can be generated by at most two elements [11, Corollary 2 to Theorem 4], but for our purposes we need to know some properties of these generators.

Lemma III.1. *Let $A = \sum_{i=1}^t n_i Q_i$ be a divisor of F and write $\mathfrak{a} = \sum_i \deg Q_i$. Then $\text{Я}(A)$ can be generated as a Я-module by two elements a_1 and a_2 satisfying $\delta_A(a_1) \leqslant 2g - 1 - \deg(A) + \mathfrak{a}$ and $\delta_A(a_2) \leqslant 4g - 2 - \deg(A) + \mathfrak{a}$.*

Proof. Prime ideals of Я correspond exactly to places of F distinct from P_∞. Therefore, from the proof of Corollary 2 to Theorem 4 in [11], we see that two elements $a_1, a_2 \in \text{Я}(A)$ generate $\text{Я}(A)$ as Я-module if and only if for all places $Q_i \in \operatorname{supp}(A)$ distinct from P_∞, we have $\min\{v_{Q_i}(a_1), v_{Q_i}(a_2)\} = -n_i$ and for any other place $Q \neq P_\infty$ of F we have $\min\{v_Q(a_1), v_Q(a_2)\} = 0$. We will construct two such elements.

Write $m_1 = 2g - 1 - \deg(A) + \mathfrak{a}$. For $j = 1, \ldots, t$, choose $a_1^{(j)} \in \mathcal{L}(A - \sum_{i\neq j} Q_i + m_1 P_\infty) \backslash \mathcal{L}(A - \sum_i Q_i + m_1 P_\infty)$. Note that such $a_1^{(j)}$ exist, since by the Riemann-Roch theorem, $l(A - \sum_{i\neq j} Q_i + m_1 P_\infty) > l(A - \sum_i Q_i + m_1 P_\infty)$. Defining $a_1 = \sum_{j=1}^t a_1^{(j)}$, we see that $v_{Q_i}(a_1) = -n_i$ for $j = 1, \ldots, t$, while $v_Q(a_1) \geqslant 0$ for any other place Q distinct from P_∞. In particular $a_1 \in \mathcal{L}(A + m_1 P_\infty)$, whence $\delta_A(a_1) \leqslant m_1$.

Now suppose that $Q_{t+1}, \ldots, Q_{t+s}$ are the zeroes of a_1 not in $\operatorname{supp}(A) \cup \{P_\infty\}$. Since $a_1 \in \mathcal{L}(A + m_1 P_\infty)$, we see that $\sum_{i=t+1}^{t+s} \deg(Q_i) \leqslant \deg(A) + m_1$. Now define $m_2 = 2g - 1 + m_1 = 4g - 2 - \deg(A) + \mathfrak{a}$. Similarly as above, we can

construct $a_2 \in \mathcal{L}(A + m_2 P_\infty)$, such that $v_{Q_i}(a_2) = 0$ for $i = t+1, \dots, t+s$. By construction $\delta_A(a_2) \leqslant m_2$. For $i = 1, \dots, t$, we have $v_{Q_i}(a_2) \geqslant -n_i$ and $v_{Q_i}(a_1) = -n_i$, whence $\min\{v_{Q_i}(a_1), v_{Q_i}(a_2)\} = -n_i$. If $Q \notin \operatorname{supp}(A) \cup \{P_\infty\}$ is not a zero of a_1, then $\min\{v_Q(a_1), v_Q(a_2)\} = 0$, since $v_Q(a_2) \geqslant 0$. If $Q \notin \operatorname{supp}(A) \cup \{P_\infty\}$ is a zero of a_1, then $v_Q(a_2) = 0$, so that also in this case $\min\{v_Q(a_1), v_Q(a_2)\} = 0$. Hence a_1 and a_2 as constructed above, generated $\text{Я}(A)$ as a Я-module. □

As $x \in \text{Я} \backslash \mathbb{F}_q$, we can also view $\text{Я}(A)$ as a free $\mathbb{F}_q[x]$-module. Following [24], we consider a special set of generators of $\text{Я}(A)$ as $\mathbb{F}_q[x]$-module, which they called the Apéry system of $\text{Я}(A)$.

Definition III.2. *For any divisor A let $y_i^{(A)} \in \mathcal{A}_i$ be such that $\delta_A(y_i^{(A)}) \leqslant \delta_A(a)$ for all $a \in \mathcal{A}_i$, where $i = 0, \dots, \mu - 1$ and*

$$\mathcal{A}_i = \{a \in \text{Я}(A) \mid \delta_A(a) \equiv i \mod \mu\} .$$

We also define $y_i = y_i^{(0)}$.

Lemma III.3. *For any divisor A it holds that*

1) *$y_0^{(A)}, \dots, y_{\mu-1}^{(A)}$ is an $\mathbb{F}_q[x]$-basis of $\text{Я}(A)$ and*
2) *$-\deg A \leqslant \delta_A(y_i^{(A)}) \leqslant 2g - 1 - \deg(A) + \mu$ for $i = 0, \dots, \mu - 1$.*

Proof. The first statement is from [24]. For the convenience of the reader we give a proof. From the strict triangle inequality for v_{P_∞}, it is clear that the elements $y_0^{(A)}, \dots, y_{\mu-1}^{(A)}$ are linearly independent over $\mathbb{F}_q[x]$. Also, it is clear that $\mathcal{Y} \subseteq \text{Я}(A)$, where $\mathcal{Y} = \langle y_0^{(A)}, \dots, y_{\mu-1}^{(A)} \rangle_{\mathbb{F}_q[x]}$. If $\mathcal{Y} \neq \text{Я}(A)$, then there would exist $a \in \text{Я}(A) \backslash \mathcal{Y}$, such that $\delta_A(a) > -\infty$ is minimal. Write $\delta_A(a) = m\mu + r$ and $\delta_A(y_r^{(A)}) = m'\mu + r$, where $m, m', r \in \mathbb{Z}$ with $0 \leqslant r < \mu$. Note that $m' \leqslant m$ by definition of $y_r^{(A)}$. Since

$$\begin{aligned}\delta_A(x^{m-m'} y_r^{(A)}) &= \delta(x^{m-m'}) + \delta_A(y_r^{(A)}) \\ &= (m - m')\mu + (m'\mu + r) = \delta_A(a) ,\end{aligned}$$

there exists a constant $\beta \in \mathbb{F}_q$ such that $\delta_A(c) < \delta_A(a)$, where $c = a - \beta x^{m-m'} y_r^{(A)} \in \text{Я}(A)$. The minimality of $\delta_A(a)$ guarantees that $c \in \mathcal{Y}$, however, this would imply that $a = c + \beta x^{m-m'} y_r^{(A)} \in \mathcal{Y}$. Hence $\mathcal{Y} = \text{Я}(A)$, which is a contradiction.

In the second statement, the lower bound simply follows from the fact that $y_i^{(A)} \in \mathcal{L}(\delta_A(y_i^{(A)})P_\infty + A) \neq \{0\}$. For the upper bound it is sufficient to show that for every integer $m > 2g-1-\deg(A)$ there exists an $a \in \text{Я}(A)$ with $\delta_A(a) = m$. But indeed, if $m > 2g-1-\deg(A)$, then $\deg(mP_\infty + A) > 2g-1$, and so [43, Theorem 1.5.17] implies that

$$\mathcal{L}(mP_\infty + A) \neq \mathcal{L}\big((m-1)P_\infty + A\big) ,$$

which concludes the proof. □

For later use, we also state the following lemma.

Lemma III.4. *If $a = \sum_{i=0}^{\mu-1} a_i y_i^{(A)} \in \text{Я}(A)$, where $a_i \in \mathbb{F}_q[x]$ and A is a divisor, then*

$$\deg a_i \leqslant \frac{1}{\mu}(\delta_A(a) - \delta_A(y_i^{(A)})) \leqslant \frac{1}{\mu}(\delta_A(a) + \deg A) .$$

Proof. Simply observe that for $i = 0, \dots, \mu - 1$ it holds that

$$\delta_A(a) = \max_j \delta_A(a_j y_j^{(A)}) \geqslant \delta(a_i) + \delta_A(y_i^{(A)}) \geqslant \delta(a_i) - \deg A ,$$

where the equality follows from the strict triangle inequality for v_{P_∞} and second inequality is given by Lemma III.3. But then

$$\begin{aligned}\deg a_i = \delta(a_i)/\mu &\leqslant \frac{1}{\mu}(\delta_A(a) - \delta_A(y_i^{(A)})) \\ &\leqslant \frac{1}{\mu}(\delta_A(a) + \deg A) .\end{aligned}$$

□

IV. Guruswami-Sudan Decoding

In this section, we paraphrase the Guruswami-Sudan list decoding algorithm [17] for $\mathcal{C}_\mathcal{L}(D, G)$ and formulate it in terms of Я modules. For the remainder of this paper fix $s, \ell \in \mathbb{Z}_{>0}$, $s \leqslant \ell$, where s is the multiplicity parameter and ℓ the designed list size of the Guruswami-Sudan list decoder. The corresponding list decoding radius will be denoted by τ.

Definition IV.1. *Let P be a rational place of F, $r \in \mathbb{F}_q$ and $Q \in F[z]$. We will say that "Q has a root of multiplicity s at (P, r)" if for any local parameter ϕ of P, there exist $c_{a,b} \in \mathbb{F}_q$ such that*

$$Q = \sum_{\substack{a,b \geqslant 0 \\ a+b \geqslant s}} c_{a,b} \phi^a (z - r)^b$$

with $c_{a,s-a} \neq 0$ for at least one $0 \leqslant a \leqslant s$.

A consequence of this definition is the following:

Lemma IV.2. *If $Q \in F[z]$ has a root of multiplicity s at (P, r) and $f \in F$ is such that $f(P) = r$, then $v_P(Q(f)) \geqslant s$.*

Proof. Writing

$$Q(f) = \sum_{\substack{a,b \geqslant 0 \\ a+b \geqslant s}} c_{a,b} \phi^a (f - r)^b ,$$

where ϕ is any local parameter of P and $c_{a,b} \in \mathbb{F}_q$, the triangle inequality directly implies that

$$\begin{aligned}v_P(Q(f)) \geqslant \min_{\substack{a,b \geqslant 0 \\ a+b \geqslant s}} \big(v_P(\phi^a) + v_P((f-r)^b)\big) &\geqslant \min_{\substack{a,b \geqslant 0 \\ a+b \geqslant s}} (a+b) \\ &\geqslant s .\end{aligned}$$

□

For any $Q = \sum_{t=0}^{\ell} z^t Q^{(t)}$ with $Q^{(t)} \in \text{Я}(-tG)$ we define $\delta_G(Q) = \max_t \delta_{-tG}(Q^{(t)})$. Moreover, for a given received word $\boldsymbol{r} = (r_1, \dots, r_n) \in \mathbb{F}_q^n$, we write

$$\mathcal{M}_{s,\ell}(D, G) = \{Q = \sum_{t=0}^{\ell} z^t Q^{(t)} \in F[z] \mid Q^{(t)} \in \text{Я}(-tG), \\ Q \text{ has a root of multiplicity at least } s \text{ at } (P_j, r_j) \text{ for all } j\}. \tag{IV.1}$$

Theorem IV.3 (Special case of Guruswami–Sudan [17])**.** *Let $\boldsymbol{r}$ be a received word and $Q \in \mathcal{M}_{s,\ell}(D, G)$ with $\delta_G(Q) < s(n-$*

τ). *If $f \in \mathcal{L}(G)$ such that the Hamming weight of $\boldsymbol{r} - \mathrm{ev}_D(f)$ is at most τ, then $Q(f) = 0$.*

Proof. Since $f^t \in \mathcal{L}(tG)$ and $Q^{(t)} \in \text{Я}(-tG)$, then $f^t Q^{(t)} \in \text{Я}$, and consequently $Q(f) \in \text{Я}$. Furthermore, since $\delta_{tG}(f^t) \leqslant 0$, then by the triangle inequality

$$\delta(Q(f)) \leqslant \max_t \delta_{-tG}(Q^{(t)}) = \delta_G(Q) \ .$$

We write $\mathcal{E} = \{j \mid r_j \neq f(P_j)\}$. Note that the cardinality of $\mathcal{E}$ is at most τ. Since $f(P_j) = r_j$ for $j \notin \mathcal{E}$, it follows from Lemma IV.2 that $Q(f) \in \text{Я}(-T)$, where $T = s\sum_{j\notin\mathcal{E}} P_j$. Since $\delta_G(Q) < s(n-\tau) \leqslant \deg T$, we may conclude that

$$Q(f) \in \mathcal{L}\big(\delta_G(Q)P_\infty - T\big) = \{0\} \ .$$

□

A. Structure of $\mathcal{M}_{s,\ell}(D,G)$ as a Я-module

The set $\mathcal{M}_{s,\ell}(D,G)$ introduced in equation (IV.1) is easily seen to be a module over the ring Я. In this subsection, we determine some of its structural properties. For the remainder of this article let $G_t = -tG - \max\{0, s-t\}D$ for $t = 0, \ldots, \ell$.

Theorem IV.4. *Let $\boldsymbol{r} = (r_1, \ldots, r_n)$ be a received word and $R \in \text{Я}(G)$ be such that $R(P_j) = r_j$ for $j = 1, \ldots, n$. Then it holds that*

$$\mathcal{M}_{s,\ell}(D,G) = \bigoplus_{t=0}^{\ell} (z-R)^t \text{Я}(G_t) \ .$$

Proof. Note that for all j and all $h \in \text{Я}(G_t)$, $v_{P_j}(h) \geqslant \max\{0, s-t\}$. Further $(z-R)^t$ has a root of multiplicity t at (P_j, r_j), since $R(P_j) = r_j$. Hence any element in $(z-R)^t\text{Я}(G_t)$ has a root of multiplicity at least s at (P_j, r_j). Moreover, since $R \in \text{Я}(G)$, we see that $(z-R)^t\text{Я}(G_t) = \left(\sum_{u=0}^t z^u \binom{t}{u}(-R)^{t-u}\right)\text{Я}(G_t) \subseteq \bigoplus_{u=0}^t z^u \text{Я}(G_u)$. Hence $(z-R)^t\text{Я}(G_t) \subseteq \mathcal{M}_{s,\ell}(D,G)$. Since $\mathcal{M}_{s,\ell}(D,G)$ is a Я-module, this implies that $\bigoplus_{t=0}^{\ell}(z-R)^t\text{Я}(G_t) \subseteq \mathcal{M}_{s,\ell}(D,G)$.

We will prove the reverse inclusion $\mathcal{M}_{s,\ell}(D,G) \subseteq \bigoplus_{t=0}^{\ell}(z-R)^t\text{Я}(G_t)$ by induction on s. Let $Q = \sum_{t=0}^{\ell} z^t Q^{(t)} \in \mathcal{M}_{s,\ell}(D,G)$ and write $Q = \sum_{t=0}^{\ell}(z-R)^t\tilde{Q}^{(t)}$ for certain $\tilde{Q}^{(t)} \in F$. Writing $z^t = ((z-R)+R)^t$ and using Newton's binomium, we obtain

$$\tilde{Q}^{(t)} = \sum_{u=t}^{\ell} \binom{u}{t} R^{u-t} Q^{(u)} \in \text{Я}(-tG), \text{ for } t = 0, \ldots, \ell,$$

since $R \in \text{Я}(G)$. Now observe that Lemma IV.2 implies that $\tilde{Q}^{(0)} = Q(R) \in \text{Я}(G_0)$.

Now if we assume $s = 1$, then $\text{Я}(G_t) = \text{Я}(-tG)$ for $t > 0$ and we can conclude from the above that $Q \in \bigoplus_{t=0}^{\ell}(z-R)^t\text{Я}(G_t)$.

If $s > 1$, we proceed as follows: using $\tilde{Q}^{(0)} \in \text{Я}(G_0) \subseteq \mathcal{M}_{s,\ell}(D,G)$, we conclude

$$\mathcal{M}_{s,\ell}(D,G) \ni Q - \tilde{Q}^{(0)} = (z-R)\cdot\sum_{t=0}^{\ell-1}(z-R)^t\tilde{Q}^{(t+1)} \ .$$

Since $z-R$ has a root of multiplicity one at (P_j, r_j) for all j, we see that $\sum_{t=0}^{\ell-1}(z-R)^t\tilde{Q}^{(t+1)}$ has a root of multiplicity at least $s-1$ at (P_j, r_j) for all j. Hence $\sum_{t=0}^{\ell-1}(z-R)^t\tilde{Q}^{(t+1)} \in \mathcal{M}_{s-1,\ell}(D,G)$. Then using the induction hypothesis for $s-1$, we may conclude that $Q \in \bigoplus_{t=0}^{\ell}(z-R)^t\text{Я}(G_t)$. □

Corollary IV.5 (of Theorem IV.4 and Lemma III.1)**.** *It holds that $\mathcal{M}_{s,\ell}(D,G) = \langle B_v^{(u)} | u = 0, \ldots, \ell,\ v = 1,2\rangle_{\text{Я}}$, where*

$$B_v^{(u)} = (z-R)^u g_v^{(u)} = \sum_{r=0}^{u}\binom{u}{r} z^r(-R)^{u-r} g_v^{(u)} \in \bigoplus_{t=0}^{\ell} z^t\text{Я}(-tG),$$

with $g_1^{(u)}, g_2^{(u)} \in \text{Я}(G_u)$ such that $\langle g_1^{(u)}, g_2^{(u)}\rangle_{\text{Я}} = \text{Я}(G_u)$,

$$\delta_{G_u}(g_1^{(u)}) \leqslant 2g - 1 + (u+1)\deg(G) + \max\{0, s-u+1\}n,$$

and

$$\delta_{G_u}(g_2^{(u)}) \leqslant 4g - 2 + (u+1)\deg(G) + \max\{0, s-u+1\}n.$$

Proof. The first part directly follows from Theorem IV.4 and Lemma III.1. To obtain the stated upper bounds on $\delta_{G_u}(g_1^{(u)})$ and $\delta_{G_u}(g_2^{(u)})$ from Lemma III.1, note that $\sum_{Q\in\mathrm{supp}(G)}\deg(Q) \leqslant \deg(G)$, since G is an effective divisor. Hence $\sum_{Q\in\mathrm{supp}(G_u)}\deg(Q) \leqslant \deg(G) + n$ if $u < s$, while $\sum_{Q\in\mathrm{supp}(G_u)}\deg(Q) \leqslant \deg(G)$ if $u \geqslant s$. The stated upper bounds are implied by this. □

Note that the proof of the corollary actually implies that for $u = s$, the stated upper bounds for $\delta_{G_u}(g_1^{(u)})$ and $\delta_{G_u}(g_2^{(u)})$ can be improved by n.

For computational purposes, we will later view $\mathcal{M}_{s,\ell}(D,G)$ as an $\mathbb{F}_q[x]$ module. Since any element from Я is an $\mathbb{F}_q[x]$-linear combination of $y_0, \ldots, y_{\mu-1}$, we obtain the following.

Corollary IV.6. *It holds that $\mathcal{M}_{s,\ell}(D,G) = \langle y_i B_v^{(u)} | i = 0, \ldots, \mu-1, u = 0, \ldots, \ell,\ v = 1,2\rangle_{\mathbb{F}_q[x]}$.*

Remark IV.7. *Since $G_t = -tG$ for $s \leqslant t \leqslant \ell$, a minor modification of the proof of Theorem IV.4 shows that $\mathcal{M}_{s,\ell}(D,G) = \bigoplus_{t=0}^{s}(z-R)^t\text{Я}(G_t) \oplus \bigoplus_{t=s+1}^{\ell}(z-R)^s z^{t-s}\text{Я}(G_t)$. This shows that the elements $\tilde{B}_v^{(u)} = B_v^{(u)}$ if $u \leqslant s$ together with $\tilde{B}_v^{(u)}(z-R)^s z^{t-s} g_v^{(u)}$ if $s < u \leqslant \ell$ form an alternative set of generators over Я for $\mathcal{M}_{s,\ell}(D,G)$. Likewise the elements in the set $\{y_i\tilde{B}_v^{(u)} | i = 0, \ldots, \mu-1, u = 0, \ldots, \ell,\ v = 1,2\}$ generate $\mathcal{M}_{s,\ell}(D,G)$ as an $\mathbb{F}_q[x]$-module. If $s < \ell$, these alternative generators can be computed using fewer operations and are therefore in general preferable.*

Remark IV.8. *The Я-module $\mathcal{M}_{s,\ell}(D,G)$ is an example of a torsion free, finitely generated module of rank $\ell+1$. Though we will not need this in the following, it interesting to note that any torsion free, finitely generated module $\mathcal{M}$ of rank r over a Dedekind domain Я, is isomorphic to a direct product of r fractional ideals of Я, say $\mathcal{M} \cong I_1 \oplus \cdots \oplus I_r$. Moreover, the product $I = I_1 \cdots I_r$ of these fractional ideals modulo principal fractional ideals only depends on the isomorphism class of $\mathcal{M}$. Therefore the element of the ideal class group of Я corresponding to I is called the Steinitz invariant of $\mathcal{M}$. See [11, Section II.4] for more details. Theorem IV.4*

can be reformulated as $\mathcal{M}_{s,\ell}(D,G) \cong \bigoplus_{u=0}^{\ell} \text{Я}(G_u)$ *and in particular the Steinitz invariant of* $\mathcal{M}_{s,\ell}(D,G)$ *is the element in the ideal class group of* Я *corresponding to*

$$\prod_{u=0}^{\ell} \text{Я}(G_u) = \text{Я}(\sum_{u=0}^{\ell} G_u) = \text{Я}(-\binom{\ell+1}{2}G - \binom{s+1}{2}D).$$

Returning to decoding, given a received word $\boldsymbol{r}$, code $\mathcal{C}_{\mathcal{L}}(D,G)$, and parameters s,ℓ, the main steps in our algorithmic approach to Guruswami-Sudan list decoding are the following.

1) Compute a generating set over Я of $\mathcal{M}_{s,\ell}(D,G)$. We will do this in Subsection V-C
2) Compute a generating set over $\mathbb{F}_q[x]$ of $\mathcal{M}_{s,\ell}(D,G)$. We will address this in Subsection V-D
3) Using fast row reduction over $\mathbb{F}_q[x]$, find a nonzero $Q \in \mathcal{M}_{s,\ell}(D,G)$ satisfying $\delta_G(Q) < s(n-\tau)$. See Subsection V-E
4) Find the roots of Q in $\mathcal{L}(G)$. See Subsection V-F

As we will see, the main result of this paper is that all these steps can be done in complexity $\tilde{\mathcal{O}}(\mu^{\omega-1}\ell^{\omega+1}(n+g))$ and with a slight variation even in $\tilde{\mathcal{O}}(\mu^{\omega-1}s\ell^{\omega}(n+g))$.

To simplify the description of the algorithms in the next sections, it will be convenient to assume that apart from P_∞, the function field F contains an additional $Z := \deg G + \max\{(\ell+1)\deg G + 4g + (s+1)n, \deg G + (\ell+3)(2g-1) + (s+1)n + 2 + \mu\}$ rational places. Even though this will not be the case in general, the same trick as at the end of Section II, will allow us to assume this. More precisely, the function field $F\mathbb{F}_{q^e}$ with $e = \lceil 2\log_q(\max\{Z, 2g+1\})\rceil$ will contain at least $1+Z$ rational places by Lemma II.3. Since using equation (II.1), $e \in \mathcal{O}(\log_q(\ell(n+g)))$, this does not interfere with our target complexity and hence does not result in any loss of generality. We will suppress the exponent e from the notation and will from now on write $\mathbb{F}_q$ for the finite field we work over, but assume that F contains all the rational places that we need to run the algorithms we describe in the next section (specifically: Algorithm 3 and Algorithm 5).

V. Algorithms

In this section, we present the algorithms that we will use to execute the Guruswami-Sudan list decoder. We start with discussing multi-point evaluation and interpolation algorithms that will form the backbone of the algorithms discussed later in the section.

A. Multi-Point Evaluation

When defining $\mathcal{C}_{\mathcal{L}}(D,G)$, we used the evaluation map ev_D. We will later need to be able to compute $\mathrm{ev}_D(f)$ fast, meaning we want to be able to evaluate the function $f \in \mathcal{L}(G)$ in the multiple points $P_1,\dots,P_n$ fast. As a matter of fact, since we will need a slightly more general setting later on, we phrase the results in this and the next subsection in terms of a very similar evaluation map, but avoid to use the divisors D and G.

Lemma V.1. *Let A be a divisor and $E = E_1 + \cdots + E_N$ for distinct rational places $E_1,\dots,E_N$ of F such that* $\mathrm{supp}(A) \cap \mathrm{supp}(E) = \varnothing$. *Further denote by* $\mathrm{ev}_E : \mathcal{L}(A) \to \mathbb{F}_q^N$ *the evaluation map defined by* $\mathrm{ev}_E(a) = (a(E_1),\dots,a(E_N))$. *Then*

1) ev_E *is injective when* $\deg A < \deg E$,
2) ev_E *is surjective when* $\deg A \geqslant \deg E + 2g - 1$.

Proof. For the first item, simply observe that the dimension of the kernel of ev_E is $l(A-E) = 0$, since $\deg(A-E) < 0$.

For the second item, observe that the dimension of the image of ev_E is

$$\begin{aligned} l(A) - l(A-E) &= \deg A - g + 1 - (\deg A - \deg E - g + 1) \\ &= \deg E \ , \end{aligned}$$

since $\deg(A) \geqslant 2g-1$ and $\deg(A-E) \geqslant 2g-1$, see [43, Theorem 1.5.17]. □

Now we state Algorithm 1, which computes $\mathrm{ev}_E(a)$ using the representation of function field elements as introduced in Section III.

Algorithm 1 Evaluate$(a, E, A, \boldsymbol{x}, \boldsymbol{y})$

Input:
- Divisors A and $E = E_1 + \cdots + E_N$, where $E_1,\dots,E_N \in \mathbb{P}_F \backslash \{P_\infty\}$ are distinct rational places and $\mathrm{supp}(A) \cap \mathrm{supp}(E) = \varnothing$,
- a function $a = \sum_{i=0}^{\mu-1} a_i y_i^{(A)} \in \text{Я}(A)$, where $a_i \in \mathbb{F}_q[x]$,
- evaluations $\boldsymbol{x} = (x_j)_{j=1,\dots,N}$, where $x_j = x(E_j) \in \mathbb{F}_q$,
- evaluations $\boldsymbol{y} = (y_{i,j})_{j=1,\dots,N}^{i=0,\dots,\mu-1}$, where $y_{i,j} = y_i^{(A)}(E_j) \in \mathbb{F}_q$.

Output:
- Evaluations $\mathrm{ev}_E(a) \in \mathbb{F}_q^N$.

1: **for** $i = 0,\dots,\mu-1$ **do**
2: $\quad (a_{i,1},\dots,a_{i,N}) \in \mathbb{F}_q^N \leftarrow (a_i(x_1),\dots,a_i(x_N))$
$\quad$ ▷ Univariate MPE
3: **return** $\sum_{i=0}^{\mu-1}(a_{i,1}y_{i,1},\dots,a_{i,N}y_{i,N}) \in \mathbb{F}_q^N$

Lemma V.2. *Algorithm 1 is correct and costs* $\tilde{\mathcal{O}}(\mu N + \delta_A(a) + \deg A)$ *operations in* $\mathbb{F}_q$.

Proof. Correctness simply follows from the fact that for $j = 1,\dots,N$

$$\begin{aligned} \sum_{i=0}^{\mu-1} a_{i,j}y_{i,j} = \sum_{i=0}^{\mu-1} a_i(x(E_j))y_i^{(A)}(E_j) &= \sum_{i=0}^{\mu-1}(a_i y_i^{(A)})(E_j) \\ &= a(E_j) \ . \end{aligned}$$

For complexity, notice that the total cost of the for-loop on Line 1 amounts to that of evaluating each of the univariate polynomials $a_0,\dots,a_{\mu-1} \in \mathbb{F}_q[x]$ on N points. According to Lemma III.4,

$$\deg a_i \leqslant \frac{1}{\mu}(\delta_A(a) + \deg A) \quad \text{for } i = 0,\dots,\mu-1 \ ,$$

hence the total cost of the for-loop is bounded by

$$\tilde{\mathcal{O}}(\mu(N + \max_i \deg a_i)) \subseteq \tilde{\mathcal{O}}(\mu N + \delta_A(a) + \deg A) \ .$$

Line 3 costs $\mathcal{O}(\mu N)$, which is subsumed by the cost of the for-loop. □

B. Interpolation

In this subsection, we address the interpolation problem. We start with an existence result.

Lemma V.3. *Let A be a divisor and $E = E_1 + \cdots + E_N$ for distinct rational places $E_1, \ldots, E_N$ of F different from P_∞ such that $\operatorname{supp}(A) \cap \operatorname{supp}(E) = \varnothing$. For any $(w_1, \ldots, w_N) \in \mathbb{F}_q^N$ there exists an $a \in \text{Я}(A)$ with*

$$\delta_A(a) \leqslant \deg E + 2g - 1 - \deg A$$

such that $a(E_j) = w_j$ for $j = 1, \ldots, N$.

Proof. Letting $A' = (\deg E + 2g - 1 - \deg A)P_\infty + A$ we get that $\deg A' \geqslant \deg E + 2g - 1$, which according to Lemma V.1 implies that the evaluation map $\mathrm{ev}_E : \mathcal{L}(A') \to \mathbb{F}_q^N$ is surjective. □

Definition V.4. *If $E = E_1 + \cdots + E_N$, where $E_1, \ldots, E_N$ are distinct rational places different from P_∞, and $U_1, \ldots, U_\mu$ are effective divisors satisfying*

1) *$E = U_1 + \cdots + U_\mu$,*
2) *$\operatorname{supp} U_i \cap \operatorname{supp} U_j = \varnothing$ when $i \neq j$,*
3) *$|\deg U_i - \deg U_j| \leqslant 1$ for all $i, j \in \{1, \ldots, \mu\}$,*
4) *for any $E_j, E_k \in \operatorname{supp} U_i$ it holds that $x(E_j) = x(E_k) \iff E_j = E_k$,*

then we will say that $U_1, \ldots, U_\mu$ is an x-partition of E.

Lemma V.5. *If $\mathcal{S}$ is a set of places such that $x(P) = x(P')$ for all $P, P' \in \mathcal{S}$, then $|\mathcal{S}| \leqslant \mu$.*

Proof. If $\alpha = x(P)$ for every $P \in \mathcal{S}$, then it is easy to see that

$$0 \neq x - \alpha \in \mathcal{L}(\mu P_\infty - \sum_{P \in \mathcal{S}} P) \ .$$

But if $\mu < |\mathcal{S}|$, then the above Riemann-Roch space has dimension zero. □

Lemma V.6. *There exists an x-partition of any divisor of the form $E = E_1 + \cdots + E_N$, where $E_1, \ldots, E_N$ are distinct rational places different from P_∞.*

Proof. We use induction on N. The base case $N = 0$ is trivial, so let us consider the induction step. Suppose $U_1, \ldots, U_\mu$ is an x-partition of $E - E_N$, and let a, b be such that U_a and U_b have minimal degree among the elements of $\{U_1, \ldots, U_\mu\}$ and $\{U_i \mid x(E_j) \neq x(E_N)$ for all $E_j \in \operatorname{supp} U_i, i = 1, \ldots, \mu\}$ respectively (b exists due to Lemma V.5). If $\deg U_a = \deg U_b$, then an x-partition of E can be obtained by replacing U_b with $U_b + E_N$. If on the other hand $\deg U_a < \deg U_b$, then U_a contains a place $\hat{E}_a$ with $x(\hat{E}_a) = x(E_N)$ and U_b contains a place $\hat{E}_b$ such that $x(\hat{E}_b) \neq x(E_j)$ for all $E_j \in U_a$. But then an x-partition of E can be obtained by replacing U_a with $U_a - \hat{E}_a + \hat{E}_b + E_N$ and U_b with $U_b - \hat{E}_b + \hat{E}_a$. □

Definition V.7. *For any polynomial matrix $\boldsymbol{A} \in \mathbb{F}_q[x]^{\phi \times \theta}$ with columns $\boldsymbol{A}_1, \ldots, \boldsymbol{A}_\theta$ and any polynomial vector $\boldsymbol{u} = (u_1, \ldots, u_\theta) \in \mathbb{F}_q[x]^\theta$ define the $\mathbb{F}_q[x]$-module*

$$\mathcal{H}_{\boldsymbol{u}}(\boldsymbol{A}) = \{\boldsymbol{v} \in \mathbb{F}_q[x]^\phi \mid \boldsymbol{v} \cdot \boldsymbol{A}_k \equiv 0 \pmod{u_k} \quad \text{for } k = 1, \ldots, \theta\} \ .$$

The following is a direct adaptation of Theorem 1.7 from [35]. We also refer to [35] for the definition of the Popov form and the $(-\boldsymbol{d})$-Popov form of a matrix. Note that if $u_1 \cdots u_\theta \neq 0$, the rank of $\mathcal{H}_{\boldsymbol{u}}(\boldsymbol{A})$ is ϕ, as $u_1 \cdots u_\theta \mathbb{F}_q[x]^\phi \subseteq \mathcal{H}_{\boldsymbol{u}}(\boldsymbol{A}) \subseteq \mathbb{F}_q[x]^\phi$. Note that the problem of computing the shifted Popov basis of $\mathcal{H}_{\boldsymbol{u}}(\boldsymbol{A})$ has been studied extensively in the literature. Earlier references than [35] are for example [20], [21]

Theorem V.8 ([35, Theorem 1.7])**.** *Assume $\phi, \theta \in \mathbb{Z}_{\geqslant 1}$ are integers such that $\phi \geqslant \theta$. There exists an algorithm which for any $\boldsymbol{A} \in \mathbb{F}_q[x]^{\phi \times \theta}$, $\boldsymbol{u} \in (\mathbb{F}_q[x] \backslash \{0\})^\theta$ and $\boldsymbol{d} = (d_1, \ldots, d_\phi) \in \mathbb{Z}_{\geqslant 0}^\phi$ can compute a matrix $\boldsymbol{V} \in \mathbb{F}_q[x]^{\phi \times \phi}$ in $(-\boldsymbol{d})$-Popov form, whose rows form an $\mathbb{F}_q[x]$-basis of $\mathcal{H}_{\boldsymbol{u}}(\boldsymbol{A})$. Furthermore, if there exists a vector $\boldsymbol{v} = (v_1, \ldots, v_\phi) \in \mathcal{H}_{\boldsymbol{u}}(\boldsymbol{A})$ satisfying the degree constraints $\deg v_t < d_t$ for $t = 1, \ldots, \phi$, then at least one row of $\boldsymbol{V}$ will also satisfy these constraints. The complexity of such an algorithm can be taken to be $\tilde{\mathcal{O}}(\phi^{\omega-1}\theta d)$ operations in $\mathbb{F}_q$, where $d = \max_t d_t + \max_k \deg u_k$.*

For our purposes, we will sometimes need to allow non-integer shifts $d_1, \ldots, d_\phi$. Non-integer, rational shifts were handled in [32] essentially by permuting columns in a very specific way:

Theorem V.9 (Reformulation of Corollary 12 in [32])**.** *Let $\boldsymbol{V} \in \mathbb{F}_q[x]^{\gamma \times \phi}$ and $\boldsymbol{d} = (d_1/\mu, \ldots, d_\phi/\mu) \in (\frac{1}{\mu}\mathbb{Z})^\phi$, where $d_1, \ldots, d_\phi \in \mathbb{Z}$. If π is the permutation on $\{1, \ldots, \phi\}$ defined by*

$$\pi(i) > \pi(j) \iff \begin{array}{c} (d_i \mod \mu) > (d_j \mod \mu) \\ \text{or} \\ (d_i \mod \mu) = (d_j \mod \mu) \quad \text{and} \quad i > j \end{array} ,$$

and $\Psi : \mathbb{F}_q[x]^\phi \to \mathbb{F}_q[x]^\phi$ is the map

$$(v_1, \ldots, v_\phi) \mapsto (x^{\lfloor d_{\pi(1)}/\mu \rfloor} v_{\pi(1)}, \ldots, x^{\lfloor d_{\pi(\phi)}/\mu \rfloor} v_{\pi(\phi)}) \ ,$$

then $\boldsymbol{V}$ is in $\boldsymbol{d}$-Popov form if and only if $\Psi(\boldsymbol{V})$ is in Popov form, where $\Psi(\boldsymbol{V}) \in \mathbb{F}_q[x]^{\gamma \times \phi}$ is the matrix created by applying Ψ to each row of $\boldsymbol{V}$.

Using the permutation defined in Theorem V.9 in combination with Theorem V.8, we obtain the following:

Corollary V.10. *In the context of Theorem V.8 we can allow $\boldsymbol{d} \in (\frac{1}{\mu}\mathbb{Z})^\phi$ and find the desired matrix $\boldsymbol{V} \in \mathbb{F}_q[x]^{\phi \times \phi}$ in complexity $\tilde{\mathcal{O}}(\phi^{\omega-1}\theta d)$ operations in $\mathbb{F}_q$, where $d = \max_t |d_t| + \max_k \deg u_k$.*

Proof. Write $\boldsymbol{d} = (\tilde{d}_1/\mu, \ldots, \tilde{d}_\phi/\mu)$ with $\tilde{d}_t \in \mathbb{Z}$ and notice that Theorem V.9 implies that $\boldsymbol{V} \in \mathbb{F}_q[x]^{\phi \times \phi}$ is in $(-\boldsymbol{d})$-Popov form if and only if $\tilde{\boldsymbol{V}} \in \mathbb{F}_q[x]^{\phi \times \phi}$ is in $(-\tilde{\boldsymbol{d}})$-Popov form, where

$$\tilde{\boldsymbol{d}} = (\lfloor \tilde{d}_{\pi(1)}/\mu \rfloor, \ldots, \lfloor \tilde{d}_{\pi(\phi)}/\mu \rfloor) \in \mathbb{Z}^\phi \ ,$$

and $\tilde{\boldsymbol{V}}$ is matrix obtained from $\boldsymbol{V}$ by permuting its columns using π from Theorem V.9. By Theorem V.8, for any matrix $\boldsymbol{A} \in \mathbb{F}_q[x]^{\phi \times \theta}$, we can compute the basis $\tilde{\boldsymbol{V}} \in \mathbb{F}_q[x]^{\phi \times \phi}$ of $\mathcal{H}_{\boldsymbol{u}}(\tilde{\boldsymbol{A}})$ in $(-\tilde{\boldsymbol{d}})$-Popov form, where $\tilde{\boldsymbol{A}} \in \mathbb{F}_q[x]^{\phi \times \theta}$ is obtained by permuting the rows of $\boldsymbol{A}$ by π, as long as the entries of $\tilde{\boldsymbol{d}}$ are non-negative. By simply adding the constant $\max_t \lceil |\tilde{d}_t|/\mu \rceil$ to all coordinates of $\tilde{\boldsymbol{d}}$, we can ensure that this is true without

breaking the target complexity. Finally, it is trivial to obtain $\boldsymbol{V}$ from $\tilde{\boldsymbol{V}}$ by applying π^{-1} to its columns. □

With these algorithmic aspects in place, we turn our attention again to the interpolation problem. We start with a lemma, which will give rise to our interpolation algorithm directly.

Lemma V.11. *Let A be a divisor and $E = E_1 + \cdots + E_N$ for distinct rational places $E_1, \ldots, E_N$ of F different from P_∞ such that $\operatorname{supp}(A) \cap \operatorname{supp}(E) = \emptyset$. Let $(w_1, \ldots, w_N) \in \mathbb{F}_q^N$ as well as an x-partition $U_1, \ldots, U_\mu$ of E be given.*

Suppose that $\boldsymbol{T} = [T_k] \in \mathbb{F}_q[x]^{1\times\mu}$ and $\boldsymbol{S} = [S_{i,k}] \in \mathbb{F}_q[x]^{\mu\times\mu}$ are such that

$$T_k(x(E_j)) = -w_j \text{ for all } E_j \in \operatorname{supp}(U_k)$$

and

$$S_{i,k}(x(E_j)) = y_i^{(A)}(E_j) \text{ for all } E_j \in \operatorname{supp}(U_k).$$

If $\boldsymbol{u} = (u_1, \ldots, u_\mu) \in \mathbb{F}_q[x]^\mu$, where

$$u_k = \prod_{E_j \in \operatorname{supp}(U_k)} (x - x(E_j)),$$

and $\boldsymbol{d} = (d_0, \ldots, d_{\mu-1}, 0) \in (\frac{1}{\mu}\mathbb{Z})^{\mu+1}$, where

$$d_i = \frac{1}{\mu}(\deg E + 2g - \deg A - \delta_A(y_i^{(A)})) \quad \text{for } i = 0, \ldots, \mu-1 ,$$

then in the $(-\boldsymbol{d})$-Popov basis of $\mathcal{H}_{\boldsymbol{u}}(\boldsymbol{A})$, where

$$\boldsymbol{A} = \left[\frac{\boldsymbol{S}}{\boldsymbol{T}}\right] \in \mathbb{F}_q^{(\mu+1)\times\mu} ,$$

there exists a vector $\boldsymbol{a} = (a_0, \ldots, a_{\mu-1}, 1) \in \mathbb{F}_q[x]^{\mu+1}$ with $\deg a_i < d_i$ for $i = 0, \ldots, \mu-1$. Moreover, if

$$a = \sum_{i=0}^{\mu-1} a_i y_i^{(A)} ,$$

then $\delta_A(a) \leqslant \deg E + 2g - 1 - \deg A$ and $a(E_j) = w_j$ for $j = 1, \ldots, N$.

Proof. Observe that according to Lemma V.3 there exists a $b \in \text{Я}(A)$ with

$$\delta_A(b) \leqslant \deg E + 2g - 1 - \deg A$$

such that $b(E_j) = w_j$ for $j = 1, \ldots, N$. If we write $b = \sum_{i=0}^{\mu-1} b_i y_i^{(A)}$, where $b_i \in \mathbb{F}_q[x]$, then it follows from Lemma III.4 that

$$\begin{aligned}\deg b_i &\leqslant \frac{1}{\mu}(\delta_A(a) - \delta_A(y_i^{(A)})) \\ &= \frac{1}{\mu}(\deg E + 2g - 1 - \deg A - \delta_A(y_i^{(A)})) < d_i .\end{aligned}$$

We claim that $\boldsymbol{b} := (b_0, \ldots, b_{\mu-1}, 1) \in \mathcal{H}_{\boldsymbol{u}}(\boldsymbol{A})$. To see this let $c_k = \sum_{i=0}^{\mu-1} b_i S_{i,k} + T_k \in \mathbb{F}_q[x]$ for $k = 1, \ldots, \mu$ and observe that for any $E_j \in U_k$ it holds that

$$c_k(x(E_j)) = \sum_{i=0}^{\mu-1} b_i(x(E_j)) y_i^{(A)}(E_j) - w_j = b(E_j) - w_j = 0 ,$$

which implies that

$$\boldsymbol{b}\boldsymbol{A}_k = c_k \equiv 0 \pmod{u_k} ,$$

where $\boldsymbol{A}_k \in \mathbb{F}_q[x]^{(\mu+1)\times 1}$ denotes the k-th column of $\boldsymbol{A}$. But then indeed $\boldsymbol{b} \in \mathcal{H}_{\boldsymbol{u}}(\boldsymbol{A})$ by definition.

Note that in the $(-\boldsymbol{d})$-degree, the leading position of $\boldsymbol{b}$ is the last position. The $(-\boldsymbol{d})$-Popov basis of $\mathcal{H}_{\boldsymbol{u}}(\boldsymbol{A})$ will contain a vector $\boldsymbol{a} = (a_0, \ldots, a_{\mu-1}, a_\mu)$ whose leading coordinate is the last position as well, and in particular $a_\mu \neq 0$. Since $\boldsymbol{a}$ has minimal $(-\boldsymbol{d})$-degree among all vectors in $\mathcal{H}_{\boldsymbol{u}}(\boldsymbol{A})$ whose leading position is the last position, we conclude that $\boldsymbol{a}$ satisfies the same degree constraints as $\boldsymbol{b}$.

To conclude the proof observe that

$$\begin{aligned}\delta_A(a) &= \max_i(\delta(a_i) + \delta_A(y_i^{(A)})) \\ &= \max_i(\mu \deg a_i + \delta_A(y_i^{(A)})) \\ &< \max_i(\mu d_i + \delta_A(y_i^{(A)})) \\ &= \deg E + 2g - \deg A ,\end{aligned}$$

and that for any $E_j \in U_k$, where $k = 1, \ldots, \mu$, it holds that

$$\begin{aligned}a(E_j) - w_j &= \sum_{i=0}^{\mu-1} a_i(x(E_j)) y_i^{(A)}(E_j) - w_j \\ &= \sum_{i=0}^{\mu-1} a_i(x(E_j)) S_{i,k}(x(E_j)) + T_k(x(E_j)) \\ &= (\boldsymbol{a}\boldsymbol{A}_k)(x(E_j)) = 0 ,\end{aligned}$$

since $\boldsymbol{a} \in \mathcal{H}_{\boldsymbol{u}}(\boldsymbol{A})$. Consequently, $a(E_j) = w_j$ for $j = 1, \ldots, N$. □

Proposition V.12. *Algorithm 2 is correct and costs $\tilde{\mathcal{O}}(\mu^{\omega-1}(N+g))$ operations in $\mathbb{F}_q$.*

Proof. Correctness is given by Lemma V.11. For complexity observe that $\deg u_k = |U_k| \leqslant \lceil N/\mu \rceil$ for all k, while for all i, k, we can choose $S_{i,k}, T_k$ such that

$$\deg S_{i,k}, \deg T_k < \lceil N/\mu \rceil .$$

Step 2 costs $\tilde{\mathcal{O}}(\mu^2 N/\mu) = \tilde{\mathcal{O}}(\mu N)$. Step 3 costs $\tilde{\mathcal{O}}(\mu N/\mu) = \tilde{\mathcal{O}}(N)$ using fast univariate interpolation [45, Corollary 10.12], and Step 4 can be executed within the same cost bound using a product tree [45, Lemma 10.4]. The computational bottleneck lies in step 6, which according to Corollary V.10 costs

$$\begin{aligned}&\tilde{\mathcal{O}}(\mu^{\omega-1}\mu(\max_i d_i + \max_k \deg u_k)) \subseteq \\ &\qquad \tilde{\mathcal{O}}(\mu^\omega(\frac{\deg E + 2g}{\mu} + \frac{N}{\mu})) = \tilde{\mathcal{O}}(\mu^{\omega-1}(N+g)) .\end{aligned}$$

Here we used that $d_i \leqslant (\deg E + 2g)/\mu$, since by Lemma III.3, $\deg A + \delta_A(y_i^{(A)}) \geqslant 0$. □

The output $a = \sum_{i=0}^{\mu-1} a_i y_i^{(A)}$ of $\mathsf{Interpolate}(\boldsymbol{w}, E, A, \boldsymbol{x}, \boldsymbol{y})$ satisfies $\delta_A(a) \leqslant \deg E + 2g - 1 - \deg A$ as shown in Lemma V.11. In general this is the best one can expect, but in specific cases the existence of an interpolation function $b \in \text{Я}(A)$ with $\delta_A(b) < \Delta < \deg E + 2g - \deg A$ may be known to exist. The following lemma clarifies a property of the output of Algorithm 2.

Algorithm 2 Interpolate$(\boldsymbol{w}, E, A, \boldsymbol{x}, \boldsymbol{y})$

Input:

- Divisors A and $E = E_1 + \cdots + E_N$, where $E_1, \ldots, E_N \in \mathbb{P}_F \setminus \{P_\infty\}$ are distinct rational places and $\operatorname{supp}(A) \cap \operatorname{supp}(E) = \varnothing$,
- interpolation values $\boldsymbol{w} = (w_1, \ldots, w_N) \in \mathbb{F}_q^N$,
- evaluations $\boldsymbol{x} = (x_j)_{j=1,\ldots,N}$, where $x_j = x(E_j) \in \mathbb{F}_q$,
- evaluations $\boldsymbol{y} = (y_{i,j})_{j=1,\ldots,N}^{i=0,\ldots,\mu-1}$, where $y_{i,j} = y_i^{(A)}(E_j) \in \mathbb{F}_q$.

Output:

- $a \in \mathcal{H}(A)$ such that $\delta_A(a) \leqslant \deg E + 2g - 1 - \deg A$ and $a(E_j) = w_j$ for $j = 1, \ldots, N$

$U_1, \ldots, U_\mu \leftarrow$ an x-partition of E
$\boldsymbol{S} = [S_{i,k}] \in \mathbb{F}_q[x]^{\mu \times \mu} \leftarrow$ matrix with $S_{i,k}(x_j) = y_{i,j}$ for all $E_j \in U_k$
$\boldsymbol{T} = [T_k] \in \mathbb{F}_q[x]^{\mu} \leftarrow$ row vector with $T_k(x_j) = -w_j$ for all $E_j \in U_k$
$\boldsymbol{u} = (u_1, \ldots, u_\mu) \in \mathbb{F}_q[x]^{\mu} \leftarrow$ vector with $u_k = \prod_{E_j \in U_k}(x - x_j)$
$\boldsymbol{d} = (d_0, \ldots, d_{\mu-1}, 1) \in (\frac{1}{\mu}\mathbb{Z})^{\mu+1} \leftarrow$ vector with $d_i = \frac{1}{\mu}(\deg E + 2g - \deg A - \delta_A(y_i^{(A)}))$
$\boldsymbol{P} \in \mathbb{F}_q[x]^{(\mu+1)\times(\mu+1)} \leftarrow (-\boldsymbol{d})$-Popov basis matrix of $\mathcal{H}_{\boldsymbol{u}}(\boldsymbol{A})$, where $\boldsymbol{A} = \left[\begin{array}{c} \boldsymbol{S} \\ \hline \boldsymbol{T} \end{array}\right] \in \mathbb{F}_q^{(\mu+1)\times\mu}$
$\boldsymbol{a} = (a_0, \ldots, a_{\mu-1}, 1) \in \mathbb{F}_q[x]^{\mu+1} \leftarrow$ a row of $\boldsymbol{P}$ having 1 as its last entry and satisfying $\deg a_i < d_i$ for $i = 0, \ldots, \mu - 1$
return $a = \sum_{i=0}^{\mu-1} a_i y_i^{(A)}$

Lemma V.13. *In the context of Algorithm 2, the output $a \in \mathcal{H}(A)$ satisfies $\delta_A(a) \leqslant \delta_A(b)$ for all functions $b \in \mathcal{H}(A)$ with $b(E_j) = w_j$ for $j = 1, \ldots, N$.*

Proof. Consider the map φ which sends any function $b = \sum_{i=0}^{\mu-1} b_i y_i^{(A)} \in \mathcal{H}(A)$ to the vector $(b_0, \ldots, b_{\mu-1}) \in \mathbb{F}_q[x]^{\mu}$, and observe that if $b(E_j) = w_j$ for all j, then $\varphi(a - b)$ is in the row space of the matrix $\tilde{\boldsymbol{P}} \in \mathbb{F}_q[x]^{\mu\times\mu}$ obtained from the first μ rows and columns of $\boldsymbol{P}$. It is clear that $\tilde{\boldsymbol{P}}$ is in $(-\tilde{\boldsymbol{d}})$-Popov form, where $\tilde{\boldsymbol{d}} = (d_0, \ldots, d_{\mu-1})$, and that each entry in $\varphi(a)$ has degree strictly smaller than the maximal degree of the corresponding column in $\tilde{\boldsymbol{P}}$: otherwise $\boldsymbol{P}$ would not be in $(-\boldsymbol{d})$-Popov form. But if each entry of $\phi(b)$ has degree no greater than the corresponding entry in $\phi(a)$, then it follows from Proposition II.8 that $\varphi(a - b) = 0$, implying that $a = b$ since $\varphi(a-b)$ is in the row space of $\tilde{\boldsymbol{P}}$ (see also [22, Theorem 6.3-15] or [44, Lemma 1.24]). □

C. Computing a generating set over $\mathcal{H}$ of $\mathcal{M}_{s,\ell}(D, G)$

We now return to the Guruswami-Sudan decoding of the code $\mathcal{C}_{\mathcal{L}}(D, G)$. In this subsection we use the symbolic expressions from Corollary IV.5 to compute a generating set over $\mathcal{H}$ of $\mathcal{M}_{s,\ell}(D, G)$. We start with a lemma.

Lemma V.14. *Let $a \in \mathcal{H}(A)$ and $b \in \mathcal{H}(B)$, where A and B are divisors, and let $E = E_1 + \cdots + E_N$, where $E_1, \ldots, E_N$ are distinct rational places different from P_∞ and not contained in $\operatorname{supp}(A) \cup \operatorname{supp}(B)$. If $c \in \mathcal{H}(A+B)$ satisfies*

1) $\delta_{A+B}(c) < N - \deg(A + B)$ and
2) $c(E_j) = a(E_j)b(E_j) = (ab)(E_j)$ for $j = 1, \ldots, N$,

then $c = ab$.

Proof. Note that $c \in \mathcal{L}(C)$, where $C = \delta_{A+B}(c)P_\infty + A + B$. The second condition simply states that $\operatorname{ev}_E(c) = \operatorname{ev}_E(ab)$, but since $\deg C < \deg E$, it follows from Lemma V.1 that $\operatorname{ev}_E : \mathcal{L}(C) \to \mathbb{F}_q^N$ is injective. Consequently, $c = ab$. □

Using Algorithm 1 and Algorithm 2, this lemma allows us to perform efficient multiplication and hence to compute a generating set over $\mathcal{H}$ of $\mathcal{M}_{s,\ell}(D, G)$ as in Algorithm 3.

Algorithm 3 Generators$_{\mathcal{H}}(\boldsymbol{r}, D, G, E, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{g})$

Input:

- Received word $\boldsymbol{r} \in \mathbb{F}_q^n$,
- the code divisors D and G,
- a divisor $E = E_1 + \cdots + E_N$, where $E_1, \ldots, E_N$ are distinct rational places of F, not in $\{P_\infty\} \cup \operatorname{supp} G$, such that $N \geqslant (\ell + 1)\deg G + 4g + (s + 1)n$,
- evaluations $\boldsymbol{x} = (x_j)_{j=1,\ldots,N}$, where $x_j = x(E_j) \in \mathbb{F}_q$,
- evaluations $\boldsymbol{y} = (y_{i,j})_{j=1,\ldots,N}^{i=0,\ldots,\mu-1}$, where $y_{i,j} = y_i^{(A)}(E_j) \in \mathbb{F}_q$,
- evaluations $\boldsymbol{g} = (g_{v,j}^{(u)})$, where $u = 0, \ldots, \ell$, $v = 1, 2$ and $j = 1, \ldots, N$ such that $g_{v,j}^{(u)} = g_v^{(u)}(E_j) \in \mathbb{F}_q$ where $\langle g_1^{(u)}, g_2^{(u)} \rangle_{\mathcal{H}} = \mathcal{H}(G_u)$, and $\delta_{G_u}(g_v^{(u)}) \leqslant 4g - 1 + (u + 1)\deg(G) + (s + 1)n$.

Output:

- $(B_v^{(u)})_{v=1,2}^{u=0,\ldots,\ell}$ such that $\langle B_v^{(u)} \rangle_{\mathcal{H}} = \mathcal{M}_{s,\ell}(D, G)$.

$R \in \mathcal{H}(G) \leftarrow$ Interpolate$(\boldsymbol{r}, D, G, \boldsymbol{x}, \boldsymbol{y})$ ▷ Algorithm 2
$(\hat{r}_1^{(0)}, \ldots, \hat{r}_N^{(0)}) \in \mathbb{F}_q^N \leftarrow (1, \ldots, 1)$
$(\hat{r}_1^{(1)}, \ldots, \hat{r}_N^{(1)}) \in \mathbb{F}_q^N \leftarrow$ Evaluate$(-R, E, G, \boldsymbol{x}, \boldsymbol{y})$ ▷ Algorithm 1
for $u = 2, \ldots, \ell$ **do**
  $(\hat{r}_1^{(u)}, \ldots, \hat{r}_N^{(u)}) \in \mathbb{F}_q^N \leftarrow (\hat{r}_1^{(1)}\hat{r}_1^{(u-1)}, \ldots, \hat{r}_N^{(1)}\hat{r}_N^{(u-1)})$
for $u = 0, \ldots, \ell$, $r = 0, \ldots, u$ and $v = 1, 2$ **do**
  $\boldsymbol{c}_{r,v}^{(u)} \in \mathbb{F}_q^N \leftarrow (\hat{r}_1^{(u-r)} g_{v,1}^{(u)}, \ldots, \hat{r}_N^{(u-r)} g_{v,N}^{(u)})$
  $c_{r,v}^{(u)} \in \mathcal{H}(-rG) \leftarrow$ Interpolate$(\boldsymbol{c}_{r,v}^{(u)}, E, -rG, \boldsymbol{x}, \boldsymbol{y})$
for $u = 0, \ldots, \ell$ and $v = 1, 2$ **do**
  $B_v^{(u)} \in \mathcal{M}_{s,\ell}(D, G) \leftarrow \sum_{r=0}^{u} \binom{u}{r} z^r c_{r,v}^{(u)}$
return $(B_v^{(u)})_{v=1,2}^{u=0,\ldots,\ell}$

Proposition V.15. *Algorithm 3 is correct and costs $\tilde{\mathcal{O}}(\ell^3 \mu^{\omega-1}(n + g))$.*

Proof. For correctness first observe that the postulated $g_v^{(u)}$ exist by Corollary IV.5.

Note that $\delta_G(R) \leqslant n + 2g - 1 - \deg G$. Using the given upper bound for $\delta_{G_u}(g_v^{(u)})$, we obtain that

$$\begin{aligned}\delta_{-rG}(R^{u-r}g_v^{(u)}) &= \delta_{(u-r)G+G_u}(R^{u-r}g_v^{(u)}) \\ &\leqslant (u-r)(n+2g-1-\deg G) + 4g - 1 \\ &\qquad + (u+1)\deg G + (s+1)n \\ &= (r+1)\deg G + (u-r+2)(2g-1) \\ &\qquad + 1 + (s+1)n \qquad \text{(V.1)} \\ &= (r+1)(\deg G - 2g + 1) \\ &\qquad + (u+3)(2g-1) + 1 + (s+1)n \\ &\leqslant (\ell+1)(\deg G - 2g + 1) \\ &\qquad + (\ell+3)(2g-1) + 1 + (s+1)n \\ &= (\ell+1)\deg G + 2(2g-1) + 1 + (s+1)n \\ &< (\ell+1)\deg G + 4g + (s+1)n.\end{aligned}$$

Lemma V.13 then implies that $\mathsf{Interpolate}(\boldsymbol{c}_{r,v}^{(u)}, E, -rG, \boldsymbol{x}, \boldsymbol{y})$ will output a function $c_{r,v}^{(u)} \in \mathcal{Y}(G)$ satisfying $\delta_{-rG}(c_{r,v}^{(u)}) < (\ell+1)\deg G + 4g + (s+1)n$.

To complete the correctness proof, we consider Lemma V.14 for the divisors $A = (u-r)G$, $B = -uG$ and the function $a = (-R)^{u-r}$, $b = g_v^{(u)}$, and $c = c_{r,v}^{(u)}$. By construction, it is clear that for all $E_j \in \operatorname{supp} E$ we have $c_{r,v}^{(u)}(E_j) = (-R)^{u-r}(E_j)g_v^{(u)}(E_j)$. Moreover, $\deg E \geqslant (\ell+1)\deg G + 4g + (s+1)n$, whence $\delta_{-rG}(c_{r,v}^{(u)}) < \deg E \leqslant \deg E - \deg(-rG)$. Hence Lemma V.14 implies $c_{r,v}^{(u)} = (-R)^{u-r}g_v^{(u)}$.

The complexity of the algorithm is dominated by the for loop in Lines 6–8. The $\mathcal{O}(\ell^2)$ calls of the algorithm $\mathsf{Interpolate}(\boldsymbol{c}_{r,v}^{(u)}, E, -rG, \boldsymbol{x}, \boldsymbol{y})$ cost $\ell^2\tilde{\mathcal{O}}(\ell\mu^{\omega-1}(n+g))$ operations. Hence the total complexity is $\tilde{\mathcal{O}}(\ell^3\mu^{\omega-1}(n+g))$. □

Remark V.16. *The generating set consisting of $\tilde{B}_v^{(u)}$ as described in Remark IV.7, can be computed slightly faster. Indeed, since in these generators, the needed powers $(-R)^u$ have the range $u = 0, \ldots, s$, the for loop in Lines 6–9 has $\mathcal{O}(s\ell)$ calls of the algorithm $\mathsf{Interpolate}(\boldsymbol{c}_{r,v}^{(u)}, E, -rG, \boldsymbol{x}, \boldsymbol{y})$. Hence to compute the $\tilde{B}_v^{(u)}$ costs $\tilde{\mathcal{O}}(s\ell^2\mu^{\omega-1}(n+g))$.*

D. Computing a generating set over $\mathbb{F}_q[x]$ of $\mathcal{M}_{s,\ell}(D,G)$.

In the previous subsection, we saw how to efficiently compute the generating set $\{B_v^{(u)}\}$ of $\mathcal{M}_{s,\ell}(D,G)$ over $\mathcal{Y}$, as in Corollary IV.5. The next logical step is to compute the set of products $\{y_iB_v^{(u)}\}$, which generates $\mathcal{M}_{s,\ell}(D,G)$ over $\mathbb{F}_q[x]$ according to Corollary IV.6. Consequently, we now consider the following problem: given a function $a \in \mathcal{Y}(A)$ for some divisor A, compute $y_0a, \ldots, y_{\mu-1}a \in \mathcal{Y}(A)$. Computing the y_ia individually using Algorithm 3 would be too slow for our purposes. Indeed, obtaining each $y_iB_v^{(u)}$ this way would cost $\tilde{\mathcal{O}}(\ell^2\mu^{\omega-1}(n+g))$ operations, and we need to compute $2\mu(\ell+1)$ such terms in total. Therefore, we introduce in this subsection a more efficient approach, which will allow us to compute $y_0a, \ldots, y_{\mu-1}a$ simultaneously.

Definition V.17. *For any $H(z) \in F[z]$ and any rational place $P \in \mathbb{P}_F$ that is not a pole of any of the coefficients of $H(z)$, and $\alpha \in \mathbb{F}_q$ we denote by $H(P,\alpha)$ the evaluation of $H(\alpha) \in F$ at P.*

Definition V.18. *Let A be a divisor and $E = E_1 + \cdots + E_N$ for distinct rational places $E_1, \ldots, E_N$ of F different from P_∞ such that $\operatorname{supp}(A) \cap \operatorname{supp}(E) = \varnothing$. For $a \in \mathcal{Y}(A)$, we define the $\mathbb{F}_q[x]$-module*

$$\mathcal{N}_{A,E}(a) = \{H = H_0 + H_1z \in \mathcal{Y}(A) \oplus z\mathcal{Y} \mid H(P, a(P)) = 0 \text{ for all } P \in \operatorname{supp}(E)\}.$$

In the following lemmas, we use the same notation A, E as in Definition V.18.

Lemma V.19. *Let $a \in \mathcal{Y}(A)$. If $H = H_0 + zH_1 \in \mathcal{N}_{A,E}(a)$ with*

$$\max\{\delta_A(H_0), \delta(H_1) + \delta_A(a)\} < \deg E - \deg A\ ,$$

then $H(a) = 0$, i.e. $H \in \langle z - a\rangle_{\mathcal{Y}}$.

Proof. Since $H \in \mathcal{N}_{A,E}(a)$, we have $H(a) \in \mathcal{Y}(A)$. Hence by definition of δ_A, we have $H(a) \in \mathcal{L}(A + \delta_A(H(a))P_\infty)$. Since for all $E_j \in \operatorname{supp} E$, we have $H(a)(E_j) = 0$ and $\operatorname{supp} E \cap (\operatorname{supp} A \cup \{P_\infty\}) = \varnothing$, we may conclude that $H(a) \in \mathcal{L}(A + \delta_A(H(a))P_\infty - E)$. Moreover,

$$\delta_A(H(a)) \leqslant \max\{\delta_A(H_0), \delta(H_1) + \delta_A(a)\} < \deg E - \deg A\ ,$$

which ensures that the aforementioned Riemann-Roch space is trivial. □

Lemma V.20. *Let $a \in \mathcal{Y}(A)$. Furthermore, let $U_1, \ldots, U_\mu$ be an x-partition of E, and let $\boldsymbol{S} = [S_{i,k}], \boldsymbol{T} = [T_{i,k}]$ be matrices in $\mathbb{F}_q[x]^{\mu\times\mu}$ such that $S_{i,k}(x(E_j)) = y_i^{(A)}(E_j)$ and $T_{i,k}(x(E_j)) = a(E_j)y_i(E_j)$ for $E_j \in U_k$. If $\boldsymbol{u} = (u_1, \ldots, u_\mu) \in \mathbb{F}_q[x]^\mu$, where $u_k = \prod_{E_j \in \operatorname{supp} U_k}(x - x(E_j))$, then the map*

$$\psi : \sum_{i=0}^{\mu-1}(s_iy_i^{(A)} + t_izy_i) \mapsto (s_0, \ldots, s_{\mu-1}, t_0, \ldots, t_{\mu-1})$$

is an $\mathbb{F}_q[x]$-isomorphism between $\mathcal{N}_{A,E}(a)$ and $\mathcal{H}_{\boldsymbol{u}}(\boldsymbol{A})$, where

$$\boldsymbol{A} = \left[\frac{\boldsymbol{S}}{\boldsymbol{T}}\right] \in \mathbb{F}_q^{2\mu\times\mu}\ .$$

Proof. Clearly ψ is an $\mathbb{F}_q[x]$-isomorphism between $\mathcal{Y}(A) \oplus z\mathcal{Y}$ and $\mathbb{F}_q[x]^{2\mu}$, therefore it suffices to show that for any $H \in \mathcal{Y}(A) \oplus z\mathcal{Y}$ it holds that $H \in \mathcal{N}_{A,E}(a)$ if and only if $\psi(H) \in \mathcal{H}_{\boldsymbol{u}}(\boldsymbol{A})$, i.e. that $H(E_j, a(E_j)) = 0$ for all $E_j \in \operatorname{supp} U_k$ and all $k = 1, \ldots, \mu$ if and only if $\psi(H) \cdot \boldsymbol{A}_k \equiv 0 \mod u_k$, for $k = 1, \ldots, \mu$, where $\boldsymbol{A}_k$ denotes the k-th column of $\boldsymbol{A}$. But this is necessarily true, since for every $E_j \in U_k$ the following identity holds, where $\alpha = x(E_j)$:

$$\begin{aligned}H(E_j, a(E_j)) &= \sum_{i=0}^{\mu-1}\left(s_i(\alpha)y_i^{(A)}(E_j) + a(E_j)t_i(\alpha)y_i(E_j)\right) \\ &= \sum_{i=0}^{\mu-1}\left(s_i(\alpha)S_{i,k}(\alpha) + t_i(\alpha)T_{i,k}(\alpha)\right) \\ &= (\psi(H) \cdot \boldsymbol{A}_k)(\alpha)\ .\end{aligned}$$

□

Lemma V.21. *In the context of Lemma V.20, if $\boldsymbol{P} \in \mathbb{F}_q[x]^{2\mu\times 2\mu}$ is the $\boldsymbol{d}$-Popov basis of $\mathcal{H}_{\boldsymbol{u}}(\boldsymbol{A}) = \psi(\mathcal{N}_{A,E}(a))$, where $\deg E \geqslant 2g + \mu + \delta_A(a) + \deg A$ and*

$$\boldsymbol{d} = \frac{1}{\mu}(\delta_A(y_0^{(A)}), \dots, \delta_A(y_{\mu-1}^{(A)}), \delta(y_0) + \delta_A(a), \dots, \delta(y_{\mu-1}) + \delta_A(a)) \in (\frac{1}{\mu}\mathbb{Z})^{2\mu} , \quad \text{(V.2)}$$

then exactly μ rows of $\boldsymbol{P}$ have $\boldsymbol{d}$-degree less than $\frac{1}{\mu}(\deg E - \deg A)$. Furthermore, if $\tilde{\boldsymbol{P}} \in \mathbb{F}_q[x]^{\mu\times 2\mu}$ is the submatrix of $\boldsymbol{P}$ consisting of these rows, then the k-th row of $\tilde{\boldsymbol{P}}$ is $\psi(Y_k)$ for $k = 1, \dots, \mu$, where $Y_k = -ay_{k-1} + zy_{k-1} \in \langle z - a\rangle_{\text{Я}} \subset \mathcal{N}_{A,E}(a)$. Consequently, if $\tilde{\boldsymbol{P}} = [\tilde{\boldsymbol{P}}_1|\tilde{\boldsymbol{P}}_2]$, where $\tilde{\boldsymbol{P}}_1, \tilde{\boldsymbol{P}}_2 \in \mathbb{F}_q[x]^{\mu\times\mu}$, then $ay_{k-1} = -\sum_{i=0}^{\mu-1} p_{k,i} y_i^{(A)}$, where $(p_{k,0}, \dots, p_{k,\mu-1})$ is the k-th row of $\tilde{\boldsymbol{P}}_1$.

Proof. For any

$$H = H_0 + zH_1 \in \mathcal{N}_{A,E}(a) ,$$

where $H_0 = \sum_{i=0}^{\mu-1} s_i y_i^{(A)} \in \text{Я}(A)$ and $H_1 = \sum_{i=0}^{\mu-1} t_i y_i \in \text{Я}$ with $s_i, t_i \in \mathbb{F}_q[x]$, it holds that

$$\begin{aligned}\deg_{\boldsymbol{d}} \psi(H) &= \max\{\max_i(\deg s_i + \frac{\delta_A(y_i^{(A)})}{\mu}), \\ &\qquad \max_i(\deg t_i + \frac{\delta(y_i) + \delta_A(a)}{\mu})\} \\ &= \frac{1}{\mu}\max\{\delta_A(H_0), \delta(H_1) + \delta_A(a)\} .\end{aligned}$$

It then follows from Lemma V.19 that

$$\deg_{\boldsymbol{d}} \psi(H) < \frac{1}{\mu}(\deg E - \deg A) \implies H \in \langle z - a\rangle_{\text{Я}} ,$$

which means that at most μ rows of $\boldsymbol{P}$ can have $\boldsymbol{d}$-degree less than $\frac{1}{\mu}(\deg E - \deg A)$, because $\langle z - a\rangle_{\text{Я}}$ has rank μ as an $\mathbb{F}_q[x]$-module. On the other hand, since $Y_1, \dots, Y_\mu$ are linearly independent over $\mathbb{F}_q[x]$, and since

$$\begin{aligned}\deg_{\boldsymbol{d}} \psi(Y_k) = \frac{1}{\mu}(\delta(y_{k-1}) + \delta_A(a)) &< \frac{1}{\mu}(\delta_A(a) + 2g + \mu) \\ &\leqslant \frac{1}{\mu}(\deg E - \deg A)\end{aligned}$$

for $k = 1, \dots, \mu$, where the strict inequality is due to Lemma III.3, then at least μ rows of $\boldsymbol{P}$ have $\boldsymbol{d}$-degree less than $\frac{1}{\mu}(\deg E - \deg A)$, because $\boldsymbol{P}$ is $\boldsymbol{d}$-row reduced. This proves the first claim of the lemma.

For the second claim it is sufficient to show that the $\boldsymbol{d}$-pivot index of $\psi(Y_k)$ is $\mu + k$, since this would imply that the matrix whose rows are $\psi(Y_k)$ is in $\boldsymbol{d}$-Popov form. To see this, write $Y_k = -\sum_{i=0}^{\mu-1} w_i y_i^{(A)} + zy_{k-1}$, where $w_i \in \mathbb{F}_q[x]$, and note that $Y_k(a) = 0$ implies that

$$\begin{aligned}\max_i \delta_A(w_i y_i^{(A)}) = \delta_A(\sum_{i=0}^{\mu-1} w_i y_i^{(A)}) &= \delta_A(ay_{k-1}) \\ &= \delta(y_{k-1}) + \delta_A(a) .\end{aligned}$$

Consequently, $\deg_{\boldsymbol{d}} \psi(Y_k) = \frac{1}{\mu}(\delta(y_{k-1}) + \delta_A(a))$, which shows that $\mu + k$ is indeed the $\boldsymbol{d}$-pivot index of $\psi(Y_k)$. □

Algorithm 4 $\mathsf{BasisProducts}_{\mathbb{F}_q[x]}(a, E, A, \boldsymbol{x}, \boldsymbol{y})$

Input:

- A divisor A,
- a function $a \in \text{Я}(A)$,
- a divisor $E = E_1 + \cdots + E_N$, where $E_1, \dots, E_N \in \mathbb{P}_F \backslash \{P_\infty\}$ are distinct rational places, $\operatorname{supp}(A) \cap \operatorname{supp}(E) = \varnothing$ and $\deg E \geqslant \deg A + \delta_A(a) + 2g + \mu$,
- evaluations $\boldsymbol{x} = (x_j)_{j=1,\dots,N}$, where $x_j = x(E_j) \in \mathbb{F}_q$,
- evaluations $\boldsymbol{y} = (y_{i,j})_{j=1,\dots,N}^{i=0,\dots,\mu-1}$, where $y_{i,j} = y_i^{(A)}(E_j) \in \mathbb{F}_q$.

Output:

- Products $(ay_0, \cdots, ay_{\mu-1})$, where each $ay_i \in \text{Я}(A)$.

1: **if** $a = 0$ **then**
2: **return** $(0, \dots, 0)$
3: $U_1, \dots, U_\mu \leftarrow$ an x-partition of E
4: $\boldsymbol{S} = [S_{i,k}] \in \mathbb{F}_q[x]^{\mu\times\mu} \leftarrow$ matrix with $S_{i,k}(x_j) = y_{i,j}$ for $E_j \in U_k$
5: $\boldsymbol{T} = [T_{i,k}] \in \mathbb{F}_q[x]^{\mu\times\mu} \leftarrow$ matrix with $T_{i,k}(x_j) = a(E_j)y_{i,j}$ for $E_j \in U_k$
6: $\boldsymbol{u} = (u_1, \dots, u_\mu) \in \mathbb{F}_q[x]^\mu \leftarrow$ vector with $u_k = \prod_{E_j \in U_k}(x - x_j)$
7: $\boldsymbol{d} \in (\frac{1}{\mu}\mathbb{Z})^{2\mu} \leftarrow \frac{1}{\mu}\big(\delta_A(y_0^{(A)}), \dots, \delta_A(y_{\mu-1}^{(A)}), \delta(y_0) + \delta_A(a), \dots, \delta(y_{\mu-1}) + \delta_A(a)\big)$
8: $\boldsymbol{P} \in \mathbb{F}_q[x]^{2\mu\times 2\mu} \leftarrow \boldsymbol{d}$-Popov basis of $\mathcal{H}_{\boldsymbol{u}}(\boldsymbol{A})$, where $\boldsymbol{A} = \left[\begin{array}{c}\boldsymbol{S}\\ \hline \boldsymbol{T}\end{array}\right] \in \mathbb{F}_q[x]^{2\mu\times\mu}$
9: $[\tilde{\boldsymbol{P}}_1|\tilde{\boldsymbol{P}}_2] \in \mathbb{F}_q[x]^{\mu\times 2\mu} \leftarrow$ the submatrix of $\boldsymbol{P}$ consisting of all rows with $\boldsymbol{d}$-degree less than $\frac{1}{\mu}(\deg E - \deg A)$, where $\tilde{\boldsymbol{P}}_1, \tilde{\boldsymbol{P}}_2 \in \mathbb{F}_q[x]^{\mu\times\mu}$
10: **for** $k = 1, \dots, \mu$ **do**
11: $(p_{k,0}, \dots, p_{k,\mu-1}) \in \mathbb{F}_q[x]^\mu \leftarrow k$-th row of $\boldsymbol{P}_1$
12: $a_k \in \text{Я}(A) \leftarrow -\sum_{i=0}^{\mu-1} p_{k,i} y_i^{(A)}$
13: **return** $(a_1, \dots, a_\mu)$

Lemma V.22. *Algorithm 4 is correct and costs $\tilde{\mathcal{O}}(\mu^{\omega-1}(N + |\deg A|))$ operations in $\mathbb{F}_q$.*

Proof. Correctness is given by Lemma V.21. For complexity, simply note that the computational bottleneck lies in Step 8, in which case $\delta_A(a) \geqslant -\deg A$ because a is nonzero and $a \in \mathcal{L}(\delta_A(a)P_\infty + A)$. By assumption, we have that $N = \deg E \geqslant \deg A + \delta_A(a) + 2g + \mu$, hence by Lemma III.3

$$\begin{aligned}-\deg A \leqslant \delta_A(y_i^{(A)}) &\leqslant 2g - 1 - \deg A + \mu \\ &< \deg E - 2\deg A - \delta_A(a) \leqslant \deg E - \deg A .\end{aligned}$$

Since $\deg u_k \leqslant N/\mu$ for $k = 1, \dots, \mu$, then the total complexity of the algorithm is given by Corollary V.10 as

$$\begin{aligned}&\tilde{\mathcal{O}}\big(\mu^{\omega-1}\max\{|\deg E|, |\deg E - \deg A|, |\deg A|\}\big) \\ &\qquad \subseteq \tilde{\mathcal{O}}(\mu^{\omega-1}(N + |\deg A|))\end{aligned}$$

operations in $\mathbb{F}_q$. □

Now we are ready to state Algorithm 5, which computes a generating set over $\mathbb{F}_q[x]$ of $\mathcal{M}_{s,\ell}(D, G)$.

Algorithm 5 $\mathsf{Generators}_{\mathbb{F}_q[x]}(\boldsymbol{r}, D, G, E, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{g})$

Input:

- Received word $\boldsymbol{r} \in \mathbb{F}_q^n$,
- divisors D and G for the code $\mathcal{C}_{\mathcal{L}}(D, G)$,
- a divisor $E = E_1 + \cdots + E_N$, where $E_1, \ldots, E_N \in \mathbb{P}_F \backslash \{P_\infty\}$ are distinct rational places, $\operatorname{supp}(A) \cap \operatorname{supp}(E) = \varnothing$ and $N \geqslant \max\{\deg G + (\ell+3)(2g-1) + (s+1)n + 2 + \mu, (\ell+1)\deg G + 4g + (s+1)n\}$,
- evaluations $\boldsymbol{x} = (x_j)_{j=1,\ldots,N}$, where $x_j = x(E_j) \in \mathbb{F}_q$,
- evaluations $\boldsymbol{y} = (y_{i,j})_{j=1,\ldots,N}^{i=0,\ldots,\mu-1}$, where $y_{i,j} = y_i^{(A)}(E_j) \in \mathbb{F}_q$,
- evaluations $\boldsymbol{g} = (g_{v,j}^{(u)})_{v=1,2,\ j=1,\ldots,N}^{u=0,\ldots,\ell}$, where $g_{v,j}^{(u)} = g_v^{(u)}(E_j) \in \mathbb{F}_q$, $\langle g_1^{(u)}, g_2^{(u)} \rangle_{\mathcal{H}} = \mathcal{H}(G_u)$ and $\delta_{G_u}(g_v^{(u)}) \leqslant 4g-1+(u+1)\deg(G)+(s+1)n$, as in Corollary IV.5.

Output:

- $(y_i B_v^{(u)})_{i=0,\ldots,\mu-1,\ v=1,2}^{u=0,\ldots,\ell}$, where $B_v^{(u)} \in \mathcal{M}_{s,\ell}(D, G)$ are as in Corollary IV.5, i.e. $\langle y_i B_v^{(u)} \rangle_{\mathbb{F}_q[x]} = \mathcal{M}_{s,\ell}(D, G)$.

1: $(B_v^{(u)})_{v=1,2}^{u=0,\ldots,\ell} \leftarrow \mathsf{Generators}_{\mathcal{H}}(\boldsymbol{r}, D, G, E, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{g})$ ▷ Algorithm 3
2: **for** $u = 0, \ldots, \ell$, $v = 1, 2$ and $t = 0, \ldots, u$ **do**
3: $\quad b_{v,t}^{(u)} \in \mathcal{H}(-tG) \leftarrow$ the z^t-coefficient of $B_v^{(u)}$
4: $\quad (y_i b_{v,t}^{(u)})_{i=0,\ldots,\mu-1} \leftarrow \mathsf{BasisProducts}_{\mathbb{F}_q[x]}(b_{v,t}^{(u)}, E, -tG, \boldsymbol{x}, \boldsymbol{y})$ ▷ Algorithm 4
5: **for** $u = 0, \ldots, \ell$, $v = 1, 2$ and $i = 0, \ldots, \mu-1$ **do**
6: $\quad B_{v,i}^{(u)} \in \mathcal{M}_{s,\ell}(D, G) \leftarrow \sum_{t=0}^{u} z^t y_i b_{v,t}^{(u)}$
7: **return** $(B_{v,i}^{(u)})_{v=1,2,\ i=0,\ldots,\mu-1}^{u=0,\ldots,\ell}$

Proposition V.23. *Algorithm 5 is correct and costs $\tilde{\mathcal{O}}(\ell^3 \mu^{\omega-1}(n+g))$ operations in $\mathbb{F}_q$.*

Proof. Correctness follows immediately from Corollary IV.6 and Lemma V.22 once we show that the calls $\mathsf{BasisProducts}_{\mathbb{F}_q[x]}(b_{v,t}^{(u)}, E, -tG, \boldsymbol{x}, \boldsymbol{y})$ in Line 4 are valid. In particular, we need to verify that

$$N \geqslant \deg(-tG) + \delta_{-tG}(b_{v,t}^{(u)}) + 2g + \mu \tag{V.3}$$

for all appropriate values of u, v and t. Using the notation from Corollary IV.5 and Algorithm 3, we know that $b_{v,t}^{(u)} = \binom{u}{t}(-R)^{u-t} g_v^{(u)}$, hence by (V.1)

$$\delta_{-tG}(b_{r,v}^{(u)}) \leqslant (t+1)\deg G + (u-t+2)(2g-1) + (s+1)n + 1 \ . \tag{V.4}$$

The sought bound (V.3) on N then follows from

$$-t\deg G + \delta_{-tG}(b_{v,t}^{(u)}) \leqslant \deg G + (\ell+2)(2g-1) + (s+1)n + 1 \ .$$

For the complexity, we note that Line 1 costs $\tilde{\mathcal{O}}(\ell^3 \mu^{\omega-1}(n+g))$ operations by Proposition V.15, while each call $\mathsf{BasisProducts}_{\mathbb{F}_q[x]}(b_{v,t}^{(u)}, E, -tG, \boldsymbol{x}, \boldsymbol{y})$ in Line 4 costs $\tilde{\mathcal{O}}(\mu^{\omega-1}(N + |\deg(-tG)|)) \subseteq \tilde{\mathcal{O}}(\ell \mu^{\omega-1}(n+g))$ operations by Lemma V.22. Since the for-loop in Line 2 has $\mathcal{O}(\ell^2)$ iterations, the stated complexity follows – the rest of the algorithm is memory management and is therefore "free". □

Remark V.24. *Computing the generating set $\{y_i \tilde{B}_v^{(u)}\}$ over $\mathbb{F}_q[x]$ of $\mathcal{M}_{s,\ell}(D, G)$ can be done in $\tilde{\mathcal{O}}(s\ell^2 \mu^{\omega-1}(n+g))$, since in that case only $\mathcal{O}(s\ell)$ coefficients of the $\tilde{B}_v^{(u)}$ are nonzero.*

E. Finding a nonzero $Q \in \mathcal{M}_{s,\ell}(D, G)$ satisfying $\delta_G(Q) < s(n-\tau)$

The following lemma introduces notation that may be needed to describe the decoding algorithm.

Lemma V.25. *For any divisor A and any $a = \sum_{i=0}^{\mu-1} a_i y_i^{(A)} \in \mathcal{H}(A)$, where $a_i \in \mathbb{F}_q[x]$, let*

$$\curlyvee^{(A)}(a) = (a_0, \ldots, a_{\mu-1}) \in \mathbb{F}_q[x]^\mu \ ,$$

and for any $Q = \sum_{t=0}^{\ell} z^t Q^{(t)} \in \bigoplus_{t=0}^{\ell} z^t \mathcal{H}(-tG)$ let

$$\curlyvee_z(Q) = (\curlyvee^{(0)}(Q^{(0)}) | \curlyvee^{(-G)}(Q^{(1)}) | \cdots | \curlyvee^{(-\ell G)}(Q^{(\ell)})) \in \mathbb{F}_q[x]^{\mu(\ell+1)} \ . \tag{V.5}$$

If $B_v^{(u)} \in \bigoplus_{t=0}^{\ell} z^t \mathcal{H}(-tG)$ for $u = 0, \ldots, \ell$ and $v = 1, 2$ are as in Corollary IV.5 and

$$\boldsymbol{M}_{s,\ell} = \left[\frac{\boldsymbol{M}_{s,\ell}^{(1)}}{\boldsymbol{M}_{s,\ell}^{(2)}} \right] \in \mathbb{F}_q[x]^{2\mu(\ell+1) \times \mu(\ell+1)} \ ,$$

where for $v = 1, 2$ the matrix $\boldsymbol{M}_{s,\ell}^{(v)}$ is given by

$$\left(\left[\begin{array}{c} \curlyvee_z(y_0 B_v^{(0)}) \\ \hline \vdots \\ \hline \curlyvee_z(y_{\mu-1} B_v^{(0)}) \end{array} \right]^\top \middle| \ \cdots \ \middle| \left[\begin{array}{c} \curlyvee_z(y_0 B_v^{(\ell)}) \\ \hline \vdots \\ \hline \curlyvee_z(y_{\mu-1} B_v^{(\ell)}) \end{array} \right]^\top \right)^\top ,$$

then $\curlyvee_z$ is an $\mathbb{F}_q[x]$-isomorphism between $\mathcal{M}_{s,\ell}(D, G)$ and the row space of $\boldsymbol{M}_{s,\ell}$. Moreover, for any Q as before, it holds that $\delta_G(Q) = \mu \deg_{\boldsymbol{d}} \curlyvee_z(Q)$, where

$$\boldsymbol{d} = (\boldsymbol{d}^{(0)} | \cdots | \boldsymbol{d}^{(\ell)}) \in (\frac{1}{\mu}\mathbb{Z})^{\mu(\ell+1)}$$

with

$$\boldsymbol{d}^{(t)} = \frac{1}{\mu}\big(\delta_{-tG}(y_0^{(-tG)}), \ldots, \delta_{-tG}(y_{\mu-1}^{(-tG)})\big) \in (\frac{1}{\mu}\mathbb{Z})^\mu$$

for $t = 0, \ldots, \ell$.

Proof. Corollary IV.6 immediately implies that $\curlyvee_z$ is an $\mathbb{F}_q[x]$-isomorphism between $\mathcal{M}_{s,\ell}(D, G)$ and the row space of $\boldsymbol{M}_{s,\ell}$. Further, writing $Q^{(t)} = \sum_{i=0}^{\mu-1} Q_i^{(t)} y_i^{(-tG)}$ for $t = 0, \ldots, \ell$, where $Q_i^{(t)} \in \mathbb{F}_q[x]$, gives that

$$\begin{aligned} \delta_G(Q) &= \max_t \delta_{-tG}(Q^{(t)}) \\ &= \max_{t,i}\{\delta_{-tG}(Q_i^{(t)} y_i^{(-tG)})\} \\ &= \max_{t,i}\{\delta(Q_i^{(t)}) + \delta_{-tG}(y_i^{(-tG)})\} \\ &= \max_{t,i}\{\mu \deg Q_i^{(t)} + \delta_{-tG}(y_i^{(-tG)})\} \\ &= \mu \deg_{\boldsymbol{d}} \curlyvee_z(Q) \ . \end{aligned}$$

□

Lemma V.25 implies that we can find a nonzero $Q \in \mathcal{M}_{s,\ell}(D,G)$ satisfying $\delta_G(Q) < s(n-\tau)$, if it exists, by computing the $\boldsymbol{d}$-Popov form of the matrix $\boldsymbol{M}_{s,\ell} \in \mathbb{F}_q[x]^{2\mu(\ell+1)\times\mu(\ell+1)}$. According to Corollary II.11, this can be achieved with cost $\tilde{\mathcal{O}}(\ell^\omega \mu^\omega \deg \boldsymbol{M}_{s,\ell})$. To estimate $\deg \boldsymbol{M}_{s,\ell}$, observe that Lemma III.4 implies that

$$\deg \boldsymbol{M}_{s,\ell} \leqslant \frac{1}{\mu} \max_{i,r,v,u} \{-r \deg G + \delta(y_i) + \delta_{-rG}(b_{r,v}^{(u)})\} \ .$$

Then Lemma III.3 and inequality (V.4) imply that

$$\deg \boldsymbol{M}_{s,\ell} \leqslant \max_{r,u} \frac{6g-2+\mu+(u-r)(n+2g-1)}{\mu} + \frac{(s+1)n+\deg G}{\mu} \in \mathcal{O}(\mu^{-1}\ell(n+g)), \tag{V.6}$$

which means that we can compute the $\boldsymbol{d}$-Popov form of $\boldsymbol{M}_{s,\ell}$ within our target complexity $\tilde{\mathcal{O}}(l^{\omega+1}\mu^{\omega-1}(n+g))$.

Remark V.26. *Using the alternative generating set from Remark IV.7, we again get an improvement on the running time. In equation* (V.6)*, the expression $u-r$ corresponded to the exponent of $-R$ in the expression $\binom{u}{r}(-R)^{u-r}g_v^{(u)}$, which was the coefficient of z^r in $B_v^{(u)}$. Since the exponent of $-R$ in a coefficient of $\tilde{B}_v^{(u)}$ never exceeds s, we therefore obtain from equation* (V.6) *the improved complexity $\tilde{\mathcal{O}}(s\ell^\omega\mu^{\omega-1}(n+g))$.*

F. Root-finding

In this subsection, we consider the final computational ingredient that we will need for Guruswami-Sudan list-decoding: given a polynomial $Q(z) \in \mathcal{M}_{s,\ell}(D,G)$, compute the set $L = \{f \in \mathcal{L}(G) \mid Q(f) = 0\}$ of all roots of Q. We accomplish this by changing the representation of Q from $\bigoplus_{t=0}^{\ell} z^t \mathcal{A}(-tG)$ to $\bigoplus_{t=0}^{\ell} z^t \mathbb{F}_q[\![x]\!]$, which will allow us to use the root-finding algorithm from [29].

Let $P_0 \notin \operatorname{supp} G \cup \{P_\infty\}$ be the fixed rational place of F for which x is a local parameter. For any nonzero $h \in F$ let $\widehat{h} \in x^{v_{P_0}(h)}\mathbb{F}_q[\![x]\!]$ denote the P_0-adic power series expansion of h in x and define $\widehat{0} = 0$. Furthermore, for any $Q = \sum_t z^t Q^{(t)} \in F[z]$ let $\widehat{Q} = \sum_t z^t \widehat{Q}^{(t)}$. Recall that if $Q^{(t)} \in \mathcal{A}(-tG)$ for all t, then $\delta_G(Q) = \max_t \delta_{-tG} Q^{(t)}$. The following definition is from [29], and it describes the output of their root-finding algorithm:

Definition V.27. *If $\widehat{Q} \in \mathbb{F}_q[\![x]\!][z]$ and $\beta \in \mathbb{Z}_{\geqslant 0}$, then a* basic root set *of $\widehat{Q}$ to precision β is a set $\{(\widehat{f}_r, \alpha_r)\}_{r=1}^m \subset \mathbb{F}_q[x] \times \mathbb{Z}_{\geqslant 0}$ with $m \leqslant \deg \widehat{Q}$ such that*

1) *$\widehat{Q}(\widehat{f}_r + x^{\alpha_r} z) \equiv 0 \pmod{x^\beta}$ for $r = 1, \ldots, m$, and*
2) *$\widehat{Q}(\widehat{f}) \equiv 0 \pmod{x^\beta} \iff \widehat{f} \in \bigcup_{r=1}^m (\widehat{f}_r + x^{\alpha_r}\mathbb{F}_q[\![x]\!])$ for every $\widehat{f} \in \mathbb{F}_q[\![x]\!]$.*

Our algorithm for computing the sought roots of $Q \in \mathcal{M}_{s,\ell}(D,G)$ will fundamentally rely on the following result:

Theorem V.28 ([29, Theorem 1.2])**.** *There is an algorithm which for any $\widehat{Q} \in \mathbb{F}_q[\![x]\!][z]$ and any precision $\beta \in \mathbb{Z}_{\geqslant 0}$ computes a basic root set of $\widehat{Q}$ to precision β using $\tilde{\mathcal{O}}(\ell\beta)$ deterministic operations in $\mathbb{F}_q$, together with an extra $\tilde{\mathcal{O}}(\mathsf{R}_{\mathbb{F}_q}(\ell)\beta)$ operations, where $\mathsf{R}_{\mathbb{F}_q}(\ell)$ is the cost of finding all $\mathbb{F}_q$-roots of a degree ℓ polynomial in $\mathbb{F}_q[z]$. Here, we can choose to use a Las Vegas algorithm with $\mathsf{R}_{\mathbb{F}_q}(\ell) \in \tilde{\mathcal{O}}(\ell)$, e.g. [45, Corollary 14.16], or a deterministic one from [41] with $\mathsf{R}_{\mathbb{F}_q}(\ell) \in \tilde{\mathcal{O}}(\ell\kappa^2\sqrt{p})$, where $|\mathbb{F}_q| = p^\kappa$ for some prime p.*

In order to use Theorem V.28 in our setting, we will need to address the following:

1) how to choose the precision β,
2) how to convert $Q \in \bigoplus_{t=0}^{\ell} z^t \mathcal{A}(-tG)$ to $\widehat{Q} \in \bigoplus_{t=0}^{\ell} z^t \mathbb{F}_q[\![x]\!]$ and
3) how to obtain the roots $f \in \mathcal{L}(G)$ of Q from a basic root set of $\widehat{Q}$.

The second item in the above list is the simplest – writing $Q = \sum_{t=0}^{\ell} z^t Q^{(t)}$ with $Q^{(t)} = \sum_{i=0}^{\mu-1} Q_i^{(t)} y_i^{(-tG)}$, where $Q_i^{(t)} \in \mathbb{F}_q[x]$, we can compute $\widehat{Q} = \sum_{t=0}^{\ell} z^t \widehat{Q}^{(t)}$ by simply relying on the identity $\widehat{Q}^{(t)} = \sum_{i=0}^{\mu-1} Q_i^{(t)} \widehat{y}_i^{(-tG)}$. Assuming that we have precomputed the $\widehat{y}_i^{(-tG)} \in \mathbb{F}_q[\![x]\!]$ to sufficiently high precision, this is just basic arithmetic in $\mathbb{F}_q[x]$.

When it comes to the choice of the precision β, then there are two restrictions that ought to be considered. The first one comes from making sure that we don't return "spurious" roots, i.e. those $f \in \mathcal{L}(G)$ such that $\widehat{Q}(\widehat{f}) \equiv 0 \pmod{x^\beta}$ while $Q(f) \neq 0$. As we are about to see in the following lemma, this issue is easily avoided by choosing $\beta > \delta_G(Q)$.

Lemma V.29. *Let $Q(z) = \sum_{t=0}^{\ell} z^t Q^{(t)}$ with $Q^{(t)} \in \mathcal{A}(-tG)$, and let $f \in \mathcal{L}(G)$. If $\beta > \delta_G(Q)$ and $\widehat{Q}(\widehat{f}) \equiv 0 \pmod{x^\beta}$, then $Q(f) = 0$.*

Proof. Notice that since $f^t Q^{(t)} \in \mathcal{A}$ for all t, then $Q(f) \in \mathcal{A}$. Furthermore, since

$$\delta(f^t Q^{(t)}) = \delta_{tG}(f^t) + \delta_{-tG}(Q^{(t)}) \leqslant \delta_{-tG}(Q^{(t)}) \leqslant \delta_G(Q) \ ,$$

where the first inequality is due to $f \in \mathcal{L}(G)$, then $\delta(Q(f)) \leqslant \delta_G(Q)$. Combining this with the assumption that $\widehat{Q}(\widehat{f}) = \widehat{Q(f)} \equiv 0 \pmod{x^\beta}$, we may conclude that $Q(f) \in \mathcal{L}(\delta_G(Q)P_\infty - \beta P_0)$, and if $\beta > \delta_G(Q)$, then this Riemann-Roch space is trivial. □

The second restriction on the precision β is posed by the task of converting the truncated power series roots of $\widehat{Q}$ back to $\mathcal{L}(G)$. Indeed, a basic root set $\{(\widehat{f}_r, \alpha_r)\}_{r=1}^m$ describes each root $\widehat{f}_r \in \mathbb{F}_q[x]$ of $\widehat{Q}$ only to precision α_r, and if this α_r is too small, then there could exist two distinct functions $h_1, h_2 \in \mathcal{L}(G)$ satisfying $\widehat{h}_1 \equiv \widehat{h}_2 \equiv \widehat{f}_r \pmod{x^{\alpha_r}}$. In Lemma V.31, we will see how we can indirectly control α_r by increasing β; but first, let us show that conversion from truncated power series to $\mathcal{L}(G)$ is guaranteed to be unambiguous as long as $\alpha_r > \deg G$.

Lemma V.30. *If $\alpha > \deg G$, then for any $h \in \mathbb{F}_q[x]$ it holds that $|\mathcal{L}(G) \cap (h + x^\alpha \mathbb{F}_q[\![x]\!])| \leqslant 1$.*

Proof. If $h_1, h_2 \in \mathcal{L}(G) \cap (h + x^\alpha \mathbb{F}_q[\![x]\!])$, then $\widehat{h_1} \equiv \widehat{h_2} \equiv h \pmod{x^\alpha}$, which means that $h_1 - h_2 \in \mathcal{L}(G - \alpha P_0) = \{0\}$. □

Now we proceed by showing that the α_r from Definition V.27 can be made arbitrarily large by choosing the precision β appropriately.

Lemma V.31. *If $Q(z) = \sum_{t=0}^{\ell} z^t Q^{(t)} \neq 0$ with $Q^{(t)} \in \text{Я}(-tG)$, and if $f \in \text{Я}(G)$ satisfies $\widehat{Q}(\widehat{f}+x^\alpha z) \equiv 0 \pmod{x^\beta}$ for some $\alpha \in \mathbb{Z}$, then $\alpha \geqslant \frac{1}{\ell}(\beta - \delta_G(Q)) - \delta_G(f)$.*

Proof. We begin by defining

$$T = Q(z+f) = \sum_{t=0}^{\ell}(z+f)^t Q^{(t)} = \sum_{t=0}^{\ell}\sum_{u=0}^{t}\binom{t}{u} z^u f^{t-u} Q^{(t)} = \sum_{u=0}^{\ell} z^u T_u \ ,$$

where $T_u = \sum_{t=u}^{\ell}\binom{t}{u} f^{t-u} Q^{(t)}$. Since $f^{t-u} \in \text{Я}((t-u)G)$ and $Q^{(t)} \in \text{Я}(-tG)$, then $T_u \in \text{Я}(-uG)$. Furthermore, $x^{\alpha u}\widehat{T}_u \equiv 0 \pmod{x^\beta}$ for all u because

$$\widehat{Q}(\widehat{f}+x^\alpha z) = \widehat{T}(x^\alpha z) = \sum_{u=0}^{\ell} z^u x^{\alpha u}\widehat{T}_u \equiv 0 \pmod{x^\beta} \ .$$

Letting $r \in \{0,\dots,\ell\}$ be such that $v_{P_0}(T_r) < \infty$ is maximal, observe that

$$\alpha\ell + v_{P_0}(T_r) \geqslant \alpha r + v_{P_0}(T_r) = v_{P_0}(x^{\alpha r}T_r) \geqslant \beta \ ,$$

which implies that $\alpha \geqslant \frac{1}{\ell}(\beta - v_{P_0}(T_r))$. Finally, noting that

$$0 \neq T_r \in \mathcal{L}(\delta_{-rG}(T_r)P_\infty - rG - v_{P_0}(T_r)P_0) \ ,$$

then the sought conclusion follows from

$$\begin{aligned} v_{P_0}(T_r) &\leqslant \delta_{-rG}(T_r) - r\deg G \leqslant \delta_{-rG}(T_r) \\ &= \delta_{-rG}\Big(\sum_{t=r}^{\ell}\binom{t}{r} f^{t-r}Q^{(t)}\Big) \\ &\leqslant \max_t\{\delta_{-rG}(f^{t-r}Q^{(t)})\} \\ &\leqslant \max_t\{(t-r)\delta_G(f) + \delta_{-tG}(Q^{(t)})\} \\ &= \ell\delta_G(f) + \delta_G(Q) \ . \end{aligned}$$

□

Combining Lemma V.31 and Lemma V.30, we obtain the final restriction

$$\beta \geqslant 2\ell\deg G + s(n-\tau) \ ,$$

which ensures that unambiguous conversion from the truncated power series roots of $\widehat{Q}$ to $\mathcal{L}(G)$ is always possible. Indeed, this bound follows immediately from the fact that $\delta_G(f) \leqslant \deg G$ for all $f \in \mathcal{L}(G)$ and the assumption that $\delta_G(Q) < s(n-\tau)$. Knowing that such conversion is possible, however, is not enough – we also need to know how to actually carry it out. In the following simple lemma, we show how to do this.

Lemma V.32. *If $f \in \mathcal{L}(G)$ and $\sum_{i=0}^{\mu-1} f_i\widehat{y}_i^{(G)} \equiv \hat{f} \pmod{x^\alpha}$ for some $f_i \in \mathbb{F}_q[x]$ with $\deg f_i \leqslant -\frac{1}{\mu}\delta_G(y_i^{(G)})$ and $\alpha > \deg G$, then $\sum_{i=0}^{\mu-1} f_i y_i^{(G)} = f$.*

Proof. Since $\delta(f_i) = \mu\deg f_i$, then $\delta_G(f_i y_i^{(G)}) \leqslant \mu\deg f_i + \delta_G(y_i^{(G)}) \leqslant 0$. But then $\sum_{i=0}^{\mu-1} f_i y_i^{(G)} \in \mathcal{L}(G)$, and the conclusion follows from Lemma V.30. □

Using the notation from Definition V.7 in the context of Lemma V.32, we see that $(f_0,\dots,f_{\mu-1},1) \in \mathcal{H}_{x^\alpha}(\boldsymbol{A})$, where $\boldsymbol{A} = [\widehat{y}_0^{(G)},\cdots,\widehat{y}_{\mu-1}^{(G)},-\widehat{f}] \in \mathbb{F}_q[x]^{1\times(\mu+1)}$. Recovering $f \in \mathcal{L}(G)$ from $\widehat{f} \text{ rem } x^\alpha$ thus translates to finding a polynomial vector $\boldsymbol{f} \in \mathcal{H}_{x^\alpha}(\boldsymbol{A})$ whose rightmost entry is 1 and $\deg_{\boldsymbol{d}}\boldsymbol{f} = 0$, where

$$\boldsymbol{d} = \frac{1}{\mu}(\delta_G(y_0^{(G)}),\dots,\delta_G(y_{\mu-1}^{(G)}),0) \in (\tfrac{1}{\mu}\mathbb{Z})^{\mu+1} \ .$$

But this is easily accomplished by relying on Theorem V.8 and Corollary V.10. We conclude this subsection by presenting our root-finding approach in its entirety in Algorithm 6.

Algorithm 6 RootFinding$(D, G, Q, \widehat{\boldsymbol{y}})$

Input:
- Divisors D and G for the code $\mathcal{C}_{\mathcal{L}}(D,G)$,
- a nonzero $Q = \sum_{t=0}^{\ell} z^t Q^{(t)} \in \mathcal{M}_{s,\ell}(D,G)$ with $\delta_G(Q) < s(n-\tau)$, where $Q^{(t)} = \sum_{i=0}^{\mu-1} Q_i^{(t)} y_i^{(-tG)}$ for some $Q_i^{(t)} \in \mathbb{F}_q[x]$,
- $\widehat{\boldsymbol{y}} = (\widehat{y}_i^{(-tG)})_{i=0,\dots,\mu-1}^{t=0,\dots,\ell}$ with $\widehat{y}_i^{(-tG)} \in \mathbb{F}_q[x]$ such that $v_{P_0}(y_i^{(-tG)} - \widehat{y}_i^{(-tG)}) \geqslant \beta := 2\ell\deg G + s(n-\tau)$.

Output:
- $L = \{f \in \mathcal{L}(G) \mid Q(f) = 0\}$ with $|L| \leqslant \ell$.

1: $\widehat{Q}^{(t)} \in \mathbb{F}_q[x] \leftarrow \sum_{i=0}^{\mu-1} Q_i^{(t)}\widehat{y}_i^{(-tG)}$ for $t = 0,\dots,\ell$
2: $\widehat{Q} \in \mathbb{F}_q[x] \leftarrow \sum_{t=0}^{\ell} z^t\widehat{Q}^{(t)}$
3: $\widehat{L} \subset \mathbb{F}_q[x] \leftarrow$ all polynomials from a basic root set of $\widehat{Q}$ to precision β
4: $L \leftarrow \varnothing$
5: $\boldsymbol{d} \in (\frac{1}{\mu}\mathbb{Z})^{\mu+1} \leftarrow \frac{1}{\mu}(\delta_G(y_0^{(G)}),\dots,\delta_G(y_{\mu-1}^{(G)}),0)$
6: $\alpha \in \mathbb{Z}_{>0} \leftarrow \deg G + 1$
7: **for** $\widehat{f} \in \widehat{L}$ **do**
8: $\quad \boldsymbol{F} \in \mathbb{F}_q^{(\mu+1)\times(\mu+1)} \leftarrow \boldsymbol{d}$-Popov basis of $\mathcal{H}_{x^\alpha}([\widehat{y}_0^{(G)},\dots,\widehat{y}_{\mu-1}^{(G)},-\widehat{f}])$
9: $\quad$ **if** $\boldsymbol{F}$ contains a row $\boldsymbol{f} = (f_0,\dots,f_{\mu-1},1)$ with $\deg_{\boldsymbol{d}}\boldsymbol{f} = 0$ **then**
10: $\quad\quad L \leftarrow L \cup \{\sum_{i=0}^{\mu-1} f_i y_i^{(G)}\}$
11: **return** L

Proposition V.33. *Algorithm 6 is correct and costs $\tilde{\mathcal{O}}(\ell^2\mu^{\omega-1}(n+g))$ operations in $\mathbb{F}_q$.*

Proof. For correctness, our goal is to prove that $L = K$, where L is the output of the algorithm and $K = \{f \in \mathcal{L}(G) \mid Q(f) = 0\}$. If $\{(\widehat{f}_r,\alpha_r)\}_{r=1}^m \subset \mathbb{F}_q[x] \times \mathbb{Z}_{\geqslant 0}$ denotes the basic root set used in Line 3, i.e. $\widehat{L} = \{\widehat{f}_r\}_{r=1}^m$, then it is clear that $K \subseteq \bigcup_{r=1}^m K_r$, where $K_r = \mathcal{L}(G) \cap (\widehat{f}_r + x^{\alpha_r}\mathbb{F}_q[\![x]\!])$ and $m \leqslant \ell$. Since $\delta_G(Q) < s(n-\tau)$, $\delta_G(\widehat{h}_r) \leqslant \deg G$ and $\beta = 2\ell\deg G + s(n-\tau)$, then Lemma V.31 guarantees that $\alpha_r \geqslant \frac{1}{\ell}\big(\beta - \delta_G(Q)\big) - \delta_G(\widehat{h}) \geqslant \deg G + 1$, hence $|K_r| \leqslant 1$ by Lemma V.30. Combining this with the fact that each non-empty K_r necessarily contains an $\mathcal{L}(G)$-root of Q, as implied by Lemma V.29 because $\beta > \delta_G(Q)$, we may conclude that $K = \bigcup_{r=1}^m K_r$. But due to Lemma V.32

$$\bigcup_{r=1}^{m} K_r = L$$
$$= \Big\{ \sum_{i=0}^{\mu-1} f_i^{(r)} y_i^{(G)} \mid \sum_{i=0}^{\mu-1} f_i^{(r)} \widehat{y}_i^{(G)} \equiv \widehat{f}_r \pmod{x^\alpha},$$
$$\deg f_i^{(r)} \leqslant -\frac{1}{\mu}\delta_G(y_i^{(G)}),\ r = 1, \dots, m \Big\}.$$

For the complexity, computing the $(\ell+1)\mu$ products $Q_i^{(t)}\widehat{y}_i^{(-tG)}$ in Line 1 costs $\tilde{\mathcal{O}}(\mu\ell\beta) \subseteq \tilde{\mathcal{O}}(\ell^2\mu(n+g))$. The basic root set of $\widehat{Q}$ in Line 3 can be computed with cost $\tilde{\mathcal{O}}(\beta \deg_z(\hat{Q})) \subseteq \tilde{\mathcal{O}}(\ell^2(n+g))$ due to [29] (see Theorem V.28). Finally, the total cost of computing the $\boldsymbol{d}$-Popov bases in line 8 across all of the $\mathcal{O}(\ell)$ iterations in the surrounding for-loop is $\tilde{\mathcal{O}}(\ell\mu^{\omega-1}(n+g))$ by Corollary V.10. The claimed complexity of the algorithm follows. □

VI. Decoding $\mathcal{C}_{\mathcal{L}}(D, G)$

We are now ready to state our Guruswami-Sudan list decoding algorithm for the code $\mathcal{C}_{\mathcal{L}}(D, G)$. We will assume that the decoding algorithm has access to the following data, which may be precomputed:

1) divisor $E = E_1 + \cdots + E_N$, where $E_1, \dots, E_N$ are distinct rational places different from P_∞ not occurring in $\operatorname{supp} G$ and $N \geqslant \max\{\deg G + (\ell+3)(2g-1) + (s+1)n + 2 + \mu, (\ell+1)\deg G + 4g + (s+1)n\}$,
2) evaluations $\boldsymbol{g} = (g_{v,j}^{(u)})$, where $u = 0, \dots, \ell$, $v = 1, 2$ and $j = 1, \dots, N$, such that $g_{v,j}^{(u)} = g_v^{(u)}(E_j) \in \mathbb{F}_q$ where $\langle g_1^{(u)}, g_2^{(u)} \rangle_{\mathcal{H}} = \mathcal{H}(G_u)$, as in Corollary IV.5
3) evaluations $\boldsymbol{x} = (x_j)_{j=1,\dots,N}$, where $x_j = x(E_j) \in \mathbb{F}_q$,
4) evaluations $\boldsymbol{y} = (y_{i,j})_{j=1,\dots,N}^{i=0,\dots,\mu-1}$, where $y_{i,j} = y_i^{(A)}(E_j) \in \mathbb{F}_q$
5) polynomials $\widehat{\boldsymbol{y}} = (\widehat{y}_i^{(-tG)})_{i,t} \in \mathbb{F}_q[x]^{\mu\times(\ell+1)}$, with $i = 0, \dots, \mu-1$ and $t = -1, \dots, \ell$, polynomials in $\mathbb{F}_q[x]$ such that $v_{P_0}(y_i^{(-tG)} - \widehat{y}_i^{(-tG)}) \geqslant 2\ell \deg G + s(n-\tau)$ for all i and t,

Then the decoding algorithm becomes as described in Algorithm 7. Note that the decoding algorithm returns the functions from $\mathcal{L}(G)$ giving rise to all codewords within radius τ of the received word. Since in Line 12, the codeword corresponding to these function have been calculated, it is trivial to modify the algorithm to return these codewords instead. Combining all results from the previous section, we immediately obtain the following:

Theorem VI.1. *The Guruswami-Sudan algorithm for the AG code $\mathcal{C}_{\mathcal{L}}(D, G)$ can be carried out in complexity $\tilde{\mathcal{O}}(\ell^{\omega+1}\mu^{\omega-1}(n+g))$. Using the alternative generating set from Remark IV.7, we obtain the complexity $\tilde{\mathcal{O}}(s\ell^\omega\mu^{\omega-1}(n+g))$.*

Algorithm 7 Decode$(\boldsymbol{r}, s, \ell, D, G)$

Input:
- Received word $\boldsymbol{r} \in \mathbb{F}_q^n$,
- divisors D and G for the code $\mathcal{C}_{\mathcal{L}}(D, G)$,
- decoding parameters $s, \ell \in \mathbb{Z}_{>0}$ with $s \leqslant \ell$,
- corresponding list-decoding radius $\tau \in \mathbb{Z}_{>0}$,

Output:
- $L = \{f \in \mathcal{L}(G) \mid d(\boldsymbol{r}, \boldsymbol{c}) \leqslant \tau\}$ or FAIL

1: $(B_{v,i}^{(u)})_{v=1,2,\ i=0,\dots,\mu-1}^{u=0,\dots,\ell}$ $\leftarrow$ Generators$_{\mathbb{F}_q[x]}(\boldsymbol{r}, D, G, E, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{g})$
2: $\boldsymbol{M}_{s,\ell} \in \mathbb{F}_q[x]^{2\mu(\ell+1)\times\mu(\ell+1)} \leftarrow$ matrix based on the $B_{v,i}^{(u)}$ as in Lemma V.25
3: $\boldsymbol{B}_{s,\ell} \in \mathbb{F}_q[x]^{\mu(\ell+1)\times\mu(\ell+1)} \leftarrow$ basis matrix in (unshifted) Popov form of $\boldsymbol{M}_{s,\ell}$
4: $\boldsymbol{d} \in (\frac{1}{\mu}\mathbb{Z})^{\mu(\ell+1)} \leftarrow (\boldsymbol{d}^{(0)} | \cdots | \boldsymbol{d}^{(\ell)})$ where for $t = 0, \dots, \ell$ $\boldsymbol{d}^{(t)} = \frac{1}{\mu}(\delta_{-tG}(y_i^{(-tG)}))_{i=0}^{\mu-1} \in (\frac{1}{\mu}\mathbb{Z})^\mu$
5: $\boldsymbol{V}_{s,\ell} \in \mathbb{F}_q[x]^{\mu(\ell+1)\times\mu(\ell+1)} \leftarrow \boldsymbol{d}$-Popov form of $\boldsymbol{B}_{s,\ell}$
6: $\boldsymbol{Q} = \big((Q_i^{(0)})_{i=0}^{\mu} | \dots | (Q_i^{(\ell)})_{i=0}^{\mu}\big) \in \mathbb{F}_q[x]^{\mu(\ell+1)} \leftarrow \deg_{\boldsymbol{d}}$-minimal row of $\boldsymbol{V}_{s,\ell}$
7: **if** $\deg_{\boldsymbol{d}} \boldsymbol{Q} \geqslant s(n-\tau)$ **then**
8: **return** FAIL
9: $Q \in \bigoplus_{t=0}^{\ell} z^t \mathcal{H}(-tG) \leftarrow \sum_{t=0}^{\ell} z^t \sum_{i=0}^{\mu-1} Q_i^{(t)} y_i^{(-tG)}$
10: $L \leftarrow$ RootFinding$(D, G, Q, \widehat{\boldsymbol{y}})$
11: **for** $f \in L$ **do**
12: $\boldsymbol{c} \in \mathbb{F}_q^n \leftarrow$ Evaluate$(f, D, G, \boldsymbol{x}, \boldsymbol{y})$
13: **if** $d(\boldsymbol{r}, \boldsymbol{c}) > \tau$ **then** $L \leftarrow L \backslash \{f\}$
14: **return** L

We now give several examples comparing this result with previously known results.

A. Examples

Example VI.2. *AG codes obtained from the rational function field $\mathbb{F}_q(x)$ are known as* generalized Reed-Solomon (GRS) codes. *In this case $g = 0$ and $\mu = 1$, which specializes the complexity of Algorithm 7 to $\tilde{\mathcal{O}}(s\ell^\omega n)$ operations in $\mathbb{F}_q$. The same complexity is achieved for families of function fields having fixed small genus, e.g. those arising from elliptic curves. The best known complexity for Guruswami-Sudan list-decoding of GRS codes is $\tilde{\mathcal{O}}(s^2\ell^{\omega-1}n)$ [9].*

Example VI.3. *By definition, any maximal function field F over $\mathbb{F}_q$ attains the Hasse-Weil bound – it has exactly $N_1 = q + 1 + 2g\sqrt{q}$ rational places, where q is necessarily a square. If F is such a function field, then any place P of $F\overline{\mathbb{F}}_q$, where $\overline{\mathbb{F}}_q$ denotes the algebraic closure of $\mathbb{F}_q$, necessarily contains a positive element no larger than $\sqrt{q}$ in its Weierstrass semigroup [19, Theorem 10.6], i.e. we are guaranteed that $\mu \leqslant \sqrt{q}$ in the complexity of Algorithm 7. Furthermore, it is well known that all maximal function fields satisfy $g \leqslant \sqrt{q}(\sqrt{q}-1)/2 \in \mathcal{O}(q)$. This implies that any code of length $n \in \Omega(q)$ over such a function field can be decoded using no more that $\tilde{\mathcal{O}}(s\ell^\omega q^{(\omega-1)/2}n) \subseteq \tilde{\mathcal{O}}(s\ell^\omega n^{(\omega+1)/2})$ operations in $\mathbb{F}_q$, which is sub-quadratic in the code length. Here, and in the rest of the examples, $u \in \Omega(v)$ if and only if $v \in \mathcal{O}(u)$ for any functions $u, v : \mathbb{R}_{\geqslant 0} \to \mathbb{R}_{\geqslant 0}$.*

We obtain even better results for long codes over specific maximal function fields:

Example VI.4. *In the case of Hermitian function field $F = \mathbb{F}_{q^2}(x_1, x_2)$, where $x_2^q + x_2 = x_1^{q+1}$, we have $N_1 = q^3 + 1$ rational places and genus $g = q(q-1)/2$. The usual choice of P_∞ in one-point codes of F gives $\mu = q$. Consequently, we can decode any such code of length $n \in \Omega(q^3)$ using*

$$\tilde{\mathcal{O}}(s\ell^\omega q^{\omega-1} q^3) = \tilde{\mathcal{O}}(s\ell^\omega q^{\omega+2}) = \tilde{\mathcal{O}}(s\ell^\omega n^{(\omega+2)/3})$$

operations in $\mathbb{F}_q$. For $n = q^3$, our approach specializes to the one from [32].

Example VI.5. *The Giulietti-Korchmaros function field $\mathbb{F}_{q^6}(x_1, x_2, x_3)$ from [14], where $x_2^q + x_2 = x_1^{q+1}$ and $x_3^{q^2-q+1} = x_1^{q^2} - x_1$, is also maximal – it has $\mu \leqslant q^3$, $g = (q^5 - 2q^3 + q^2)/2$ and $N_1 = q^8 - q^6 + q^5 + 1$. In this case, we can decode any code of length $n \in \Omega(q^8)$ with cost $\tilde{\mathcal{O}}(s\ell^\omega n^{(3\omega+5)/8})$.*

Example VI.6. *The Suzuki function field $F = \mathbb{F}_q(x_1, x_2)$, where $q = 2^{2e+1}$ is an odd power of two and $x_2^q + x_2 = x_1^{2^e}(x_1^q + x_1)$, has genus $g = 2^e(q-1)$ and $N_1 = q^2 + 1$ rational places. Although it is not maximal in the sense of the Hasse-Weil bound, no other function field with the same genus and constant field can surpass its number of rational places [40, Section 5.4]. From [3], it immediately follows that the Weierstrass semigroup of any place P contains a positive element no greater than q, i.e. $\mu \leqslant q$. This means that for any code over F of length $n \in \Omega(q^2)$, the complexity of Algorithm 7 specializes to $\tilde{\mathcal{O}}(s\ell^\omega n^{(\omega+1)/2})$.*

Example VI.7. *Let F be a function field over $\mathbb{F}_q$ having a rational place P_∞ whose Weierstrass semigroup can be generated by two positive integers, say a and b, where $a < b$. Note that necessarily $\gcd(a, b) = 1$, since otherwise the semigroup generated by a and b has infinitely many gaps. The genus of such a function field is $(a-1)(b-1)/2$, since this is the number of gaps of the semigroup generated by a and b. Now let $x, y \in \mathcal{A}$ be such that $\delta(x) = a$ and $\delta(y) = b$. Then $F = \mathbb{F}_q(x, y)$ and $x^b + \alpha y^a + g(x, y) = 0$, where $\alpha \in \mathbb{F}_q \backslash \{0\}$ and $g(X, Y) \in \mathbb{F}_q[X, Y]$ has (a, b)-weighted degree strictly less then ab. The curve defined by the equation $X^b + \alpha Y^a + g(X, Y) = 0$ is sometimes called a C_{ab}-curve or a Miura-Kayima curve [28]; codes defined over such curves are of particular interest for practical applications, as they can be encoded efficiently [6]. When it comes to decoding, the additional assumptions that $G = mP_\infty$ and that $D - nP_\infty$ is a principal divisor were used in [4] to decode the code $\mathcal{C}_\mathcal{L}(D, G)$ in complexity $\tilde{\mathcal{O}}(\ell^5 a^3(n+g))$.*

Let us compare this to our results. Knowing that F has a rational point P_∞ whose Weierstrass semigroup contains two positive, relatively prime integers a and b, implies that $g \leqslant (a-1)(b-1)/2$ and $\mu \leqslant a$. Using this weaker assumption and not needing the additional requirement that $G = mP_\infty$ and that $D - nP_\infty$ is a principal divisor, we can decode $\mathcal{C}_\mathcal{L}(D, G)$ in complexity $\tilde{\mathcal{O}}(s\ell^\omega a^{\omega-1}(n+g))$. Hence, our results can both handle more general settings and decode faster.

References

[1] M. Alekhnovich. Linear Diophantine Equations Over Polynomials and Soft Decoding of Reed–Solomon Codes. *IEEE Transactions on Information Theory*, 51(7):2257–2265, July 2005.

[2] J. Alman and V. V. Williams. A refined laser method and faster matrix multiplication. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 522–539. SIAM, 2021.

[3] D. Bartoli, M. Montanucci, and G. Zini. Weierstrass semigroups at every point of the Suzuki curve. *Acta Arith.*, 197(1):1–20, 2021.

[4] P. Beelen and K. Brander. Efficient list decoding of a class of algebraic-geometry codes. *Advances in Mathematics of Communications*, 4(4):485–518, Nov. 2010.

[5] P. Beelen and T. Høholdt. The Decoding of Algebraic Geometry Codes. In E. Martínez-Moro, editor, *Advances in Algebraic Geometry Codes*, volume 5. World Scientific Publishing Company, 2008.

[6] P. Beelen, J. Rosenkilde, and G. Solomatov. Fast encoding of ag codes over cab curves. *IEEE Transactions on Information Theory*, 67(3):1641–1655, 2020.

[7] J. Berthomieu, G. Lecerf, and G. Quintin. Polynomial root finding over local rings and application to error correcting codes. *Applicable Algebra in Engineering, Communication and Computing*, 24(6):413–443, July 2013.

[8] D. G. Cantor and E. Kaltofen. On fast multiplication of polynomials over arbitrary algebras. *Acta Informatica*, 28(7):693–701, July 1991.

[9] M. Chowdhury, C.-P. Jeannerod, V. Neiger, E. Schost, and G. Villard. Faster Algorithms for Multivariate Interpolation With Multiplicities and Simultaneous Polynomial Approximations. *IEEE Transactions on Information Theory*, 61(5):2370–2387, May 2015.

[10] J. Farr and S. Gao. Grobner bases, pade approximation, and decoding of linear codes. *Contemporary Mathematics*, 381:3, 2005.

[11] A. Fröhlich, M. J. Taylor, and M. J. Taylor. *Algebraic number theory*. Number 27. Cambridge University Press, 1991.

[12] S. Gao and M. A. Shokrollahi. Computing roots of polynomials over function fields of curves. In *Coding Theory and Cryptography*, pages 214–228. Springer, 2000.

[13] P. Giorgi, C. Jeannerod, and G. Villard. On the Complexity of Polynomial Matrix Computations. In *International Symposium on Symbolic and Algebraic Computation*, pages 135–142, 2003.

[14] M. Giulietti and G. Korchmáros. A new family of maximal curves over a finite field. *Math. Ann.*, 343(1):229–245, 2009.

[15] V. D. Goppa. Algebraico-Geometric Codes. *Mathematics of the USSR-Izvestiya*, 21(1):75, 1983.

[16] V. Guruswami and M. Sudan. Improved Decoding of Reed–Solomon and Algebraic-Geometric Codes. In *IEEE Annual Symposium on Foundations of Computer Science*, pages 28–37, 1998.

[17] V. Guruswami and M. Sudan. Improved Decoding of Reed–Solomon Codes and Algebraic-Geometric Codes. *IEEE Transactions on Information Theory*, 45(6):1757–1767, 1999.

[18] T. Høholdt, J. H. van Lint, and R. Pellikaan. Algebraic geometry codes. In W. C. Huffman and V. S. Pless, editors, *Handbook of Coding Theory*. Elsevier Science Inc., 1998.

[19] J. W. P. Hirschfeld, G. Korchmáros, and F. Torres. *Algebraic curves over a finite field*. Princeton Series in Applied Mathematics. Princeton University Press, Princeton, NJ, 2008.

[20] C.-P. Jeannerod, V. Neiger, E. Schost, and G. Villard. Fast Computation of Minimal Interpolation Bases in Popov Form for Arbitrary Shifts. In *International Symposium on Symbolic and Algebraic Computation*, ISSAC '16, pages 295–302, New York, NY, USA, 2016. ACM.

[21] C.-P. Jeannerod, V. Neiger, E. Schost, and G. Villard. Computing minimal interpolation bases. *Journal of Symbolic Computation*, 83:272–314, Nov. 2017.

[22] T. Kailath. *Linear Systems*. Prentice-Hall, 1980.

[23] C. Kirfel and R. Pellikaan. The minimum distance of codes in an array coming from telescopic semigroups. volume 41, pages 1720–1732. 1995. Special issue on algebraic geometry codes.

[24] K. Lee, M. Bras-Amoros, and M. O'Sullivan. Unique Decoding of General AG Codes. *IEEE Transactions on Information Theory*, 60(4):2038–2053, Apr. 2014.

[25] K. Lee and M. E. O'Sullivan. List Decoding of Reed–Solomon Codes from a Gröbner Basis Perspective. *Journal of Symbolic Computation*, 43(9):645 – 658, 2008.

[26] K. Lee and M. E. O'Sullivan. List decoding of Hermitian codes using Gröbner bases. *Journal of Symbolic Computation*, 44(12):1662–1675, 2009.

[27] R. McEliece. The Guruswami-Sudan Decoding Algorithm for Reed-Solomon Codes. *IPN progress report*, pages 42–153, 2003.

[28] S. Miura and N. Kamiya. Geometric-goppa codes on some maximal curves and their minimum distance. *Proceedings of 1993 IEEE Information Theory Workshop*, pages 85–86, 06 1993.

[29] V. Neiger, J. Rosenkilde, and E. Schost. Fast Computation of the Roots of Polynomials Over the Ring of Power Series. In *International Symposium on Symbolic and Algebraic Computation*, July 2017.

[30] V. Neiger and T. X. Vu. Computing Canonical Bases of Modules of Univariate Relations. In *International Symposium on Symbolic and Algebraic Computation*, page 8, July 2017.

[31] H. Niederreiter and C. Xing. *Rational points on curves over finite fields: theory and applications*, volume 285 of *London Mathematical Society Lecture Note Series*. Cambridge University Press, Cambridge, 2001.

[32] J. Nielsen and P. Beelen. Sub-Quadratic Decoding of One-Point Hermitian Codes. *IEEE Transactions on Information Theory*, 61(6):3225–3240, June 2015.

[33] R. R. Nielsen and T. Høholdt. Decoding reed-solomon codes beyond half the minimum distance. In *Coding Theory, Cryptography and Related Areas*, pages 221–236. Springer, 2000.

[34] H. O'Keeffe and P. Fitzpatrick. Gröbner basis solutions of constrained interpolation problems. *Linear algebra and its applications*, 351:533–551, 2002.

[35] J. Rosenkilde and A. Storjohann. Algorithms for simultaneous hermite–padé approximations. *Journal of Symbolic Computation*, 102:279 – 303, 2021.

[36] R. Roth and G. Ruckenstein. Efficient Decoding of Reed–Solomon Codes Beyond Half the Minimum Distance. *IEEE Transactions on Information Theory*, 46(1):246 –257, 2000.

[37] S. Sakata. Extension of the Berlekamp-Massey algorithm to N dimensions. *Information and Computation*, 84(2):207–239, 1990.

[38] S. Sakata and M. Fujisawa. Fast Decoding of Multipoint Codes from Algebraic Curves. *IEEE Transactions on Information Theory*, 60(4):2054–2064, Apr. 2014.

[39] S. Sakata, H. E. Jensen, and T. Høholdt. Generalized Berlekamp-Massey Decoding of Algebraic-Geometric Codes up to Half the Feng–Rao Bound. *IEEE Transactions on Information Theory*, 41(6):1762–1768, 1995.

[40] J.-P. Serre. *Rational points on curves over finite fields*, volume 18 of *Documents Mathématiques (Paris)*.

[41] V. Shoup. A fast deterministic algorithm for factoring polynomials over finite fields of small characteristic. In *Proceedings of the 1991 international symposium on Symbolic and algebraic computation*, pages 14–21, 1991.

[42] G. Solomatov. *Computational aspects of Algebraic Geometry codes*. PhD thesis, Technical University of Denmark, 2021.

[43] H. Stichtenoth. *Algebraic Function Fields and Codes*. Springer, 2nd edition, 2009.

[44] Vincent Neiger. *Bases of relations in one or several variables: fast algorithms and applications*. PhD Thesis, ENS Lyon, Nov. 2016.

[45] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge University Press, 3rd edition, 2012.

[46] W. Zhou and G. Labahn. Computing Column Bases of Polynomial Matrices. In *International Symposium on Symbolic and Algebraic Computation*, ISSAC '13, pages 379–386, New York, NY, USA, 2013. ACM.

Peter Beelen received his Master's degree in Mathematics from the University of Utrecht, The Netherlands, in 1996. In 2001 he received his Ph.D. degree in Mathematics from the Technical University of Eindhoven, The Netherlands. Since October 2004 he has been a staff member of the Technical University of Denmark (DTU), Kongens Lyngby, Denmark. He has been an assistant professor at DTU till January 2007 and an associate professor till August 2014. Since September 2014 he has worked at DTU as professor. His research interests include various aspects of algebra and its applications, notably algebraic curves and algebraic coding theory.

Johan Rosenkilde received the master's degree in computer science and the Ph.D. degree in mathematics from the Technical University of Denmark. He has been a Research Engineer at GitHub since 2021. Before that, he was at the Technical University of Denmark, first as an Assistant Professor and later as an Associate Professor. He was a Post-Doctoral Researcher at both Ulm University, Germany, and Inria, France. His algebraic research interests include coding theory and computer algebra.

Grigory Solomatov received a dual master's degree in mathematics in 2017 from the Norwegian University of Science and Technology and the Technical University of Denmark. In 2021, he received a Ph.D degree in mathematics from the latter. Currently, he is holding a post-doctoral position at Tel Aviv University. His research interests include computer algebra and coding theory.