DTU Library

Multi-Trial Guruswami–Sudan Decoding for Generalised Reed–Solomon Codes

Nielsen, Johan Sebastian Rosenkilde; Zeh, Alexander

Published in:
WCC 2013 International Workshop on Coding and Cryptography

Publication date:
2013

Document Version
Early version, also known as pre-print

Noname manuscript No.
(will be inserted by the editor)

Multi-Trial Guruswami–Sudan Decoding for Generalised Reed–Solomon Codes

Johan S. R. Nielsen · Alexander Zeh

February 1, 2013

Abstract An iterated refinement procedure for the Guruswami–Sudan list decoding algorithm for Generalised Reed–Solomon codes based on Alekhnovich's module minimisation is proposed. The method is parametrisable and allows variants of the usual list decoding approach. In particular, finding the list of *closest* codewords within an intermediate radius can be performed with improved average-case complexity while retaining the worst-case complexity.

Keywords Guruswami–Sudan · List Decoding · Reed–Solomon Codes · Multi-Trial

1 Introduction

Since the discovery of a polynomial-time hard-decision list decoder for Generalised Reed–Solomon (GRS) codes by Guruswami and Sudan (GS) [12, 7] in the late 1990s, much work has been done to speed up the two main parts of the algorithm: interpolation and root-finding. Notably, for interpolation Beelen and Brander [2] mixed the module reduction approach by Lee and O'Sullivan [8] with the parametrisation of Zeh *et al.* [13], and employed the fast module reduction algorithm by Alekhnovich [1]. Bernstein [4] pointed out that a slightly faster variant can be achieved by using the reduction algorithm by Giorgi *et al.* [6].

For the root-finding step, one can employ the method of Roth and Ruckenstein [11] in a divide-and-conquer fashion, as described by Alekhnovich [1]. This step then becomes an order of magnitude faster than interpolation, leaving the latter as the main target for further optimisations.

For a given code, the GS algorithm has two parameters, both positive integers: the interpolation multiplicity s and the list size ℓ. Together with the code parameters they determine the decoding radius τ. To achieve a higher decoding radii for some given

J. S. R. Nielsen is with the Institute of Mathematics, Technical University of Denmark
E-mail: `j.s.r.nielsen@mat.dtu.dk`
A. Zeh is with the Institute of Communications Engineering, University of Ulm, Germany and Research Center INRIA Saclay - Île-de-France, École Polytechnique, France
E-mail: `alex@codingtheory.eu`

GRS code, one needs higher s and ℓ, and the value of these strongly influence the running time of the algorithm.

In this work, we present a novel iterative method: we first solve the interpolation problem for $s = \ell = 1$ and then iteratively refine this solution for increasing s and ℓ. In each step of our algorithm, we obtain a valid solution to the interpolation problem for these intermediate parameters. The method builds upon that of Beelen–Brander [2] and has the same asymptotic complexity.

The method therefore allows a fast *multi-trial* list decoder when our aim is just to find the list of codewords with minimal distance to the received word. At any time during the refinement process, we will have an interpolation polynomial for intermediate parameters $\hat{s} \leq s$, $\hat{\ell} \leq \ell$ yielding an intermediate decoding radius $\hat{\tau} \leq \tau$. If we perform the root-finding step of the GS algorithm on this, all codewords with distance at most $\hat{\tau}$ from the received are returned; if there are any such words, we break computation and return those; otherwise we continue the refinement. We can choose any number of these trials, e.g. for each possible intermediate decoding radius between half the minimum distance and the target τ.

Since the root-finding step of GS is cheaper than the interpolation step, this multi-trial decoder will have the same asymptotic worst-case complexity as the usual GS using the Beelen–Brander interpolation; however, the average-case complexity is better since fewer errors are more probable.

This contribution is structured as follows. In the next section we give necessary preliminaries and state the GS interpolation problem for decoding GRS codes. In Section 3 we give a definition and properties of minimal matrices. Alekhnovich's algorithm can bring matrices to this form, and we give a more fine-grained bound on its asymptotic complexity. Our new iterative procedure is explained in detail in Section 4.

2 Preliminaries

2.1 Notation

Let $\mathbb{F}_q$ be the finite field of order q and let $\mathbb{F}_q[X]$ be the polynomial ring over $\mathbb{F}_q$ with indeterminate X. Let $\mathbb{F}_q[X,Y]$ denote the polynomial ring in the variables X and Y and let $\mathrm{wdeg}_{u,v} X^iY^j \triangleq ui + vj$ be the (u,v)-weighted degree of X^iY^j.

A vector of length n is denoted by $\mathbf{v} = (v_0, \ldots, v_{n-1})$. If $\mathbf{v}$ is a vector over $\mathbb{F}_q[X]$, let $\deg \mathbf{v} \triangleq \max_i\{\deg v_i(X)\}$. We introduce the leading position as $\mathrm{LP}(\mathbf{v}) = \max_i\{i | \deg v_i(X) = \deg \mathbf{v}\}$ and the leading term $\mathrm{LT}(\mathbf{v}) = v_{\mathrm{LP}(\mathbf{v})}$ is the term at this position. An $m \times n$ matrix is denoted by $\mathcal{V} = \|v_{i,j}\|_{i=0,j=0}^{m-1,n-1}$. The rows of such a matrix will be denoted by lower-case letters, e.g. $\mathbf{v}_0, \ldots, \mathbf{v}_{m-1}$. Furthermore, let $\deg \mathcal{V} = \sum_{i=0}^{m-1} \deg \mathbf{v}_i$. Modules are denoted by capital letters such as M.

2.2 Interpolation-Based Decoding of GRS Codes

Let $\alpha_0, \ldots, \alpha_{n-1}$ be n nonzero distinct elements of $\mathbb{F}_q$ with $n < q$ and let $w_0, \ldots, w_{n-1}$ be n (not necessarily distinct) nonzero elements of $\mathbb{F}_q$. A GRS code $\mathcal{GRS}(n,k)$ of length n and dimension k over $\mathbb{F}_q$ is given by

$$\mathcal{GRS}(n,k) \triangleq \big\{(w_0 f(\alpha_0), \ldots, w_{n-1} f(\alpha_{n-1})) : f(X) \in \mathbb{F}_q[X],\, \deg f(X) < k\big\}. \quad (1)$$

GRS codes are *Maximum Distance Separable* (MDS) codes, i.e., their minimum Hamming distance is $d = n - k + 1$. We shortly explain the interpolation problem of GS [7, 12] for decoding GRS codes in the following.

Theorem 1 (Guruswami–Sudan for GRS Codes [7,12]) *Let* $\mathbf{c} \in \mathcal{GRS}(n, k)$ *be a codeword and* $f(X)$ *the corresponding information polynomial as defined in* (1). *Let* $\mathbf{r} = (r_0, \dots, r_{n-1}) = \mathbf{c} + \mathbf{e}$ *be a received word where* $\mathrm{weight}(\mathbf{e}) \le \tau$. *Let* r'_i *denote* r_i/w_i.

Let $Q(X, Y) \in \mathbb{F}_q[X, Y]$ *be a nonzero polynomial that passes through the* n *points* $(\alpha_0, r'_0), \dots, (\alpha_{n-1}, r'_{n-1})$ *with multiplicity* $s \ge 1$, *has* Y*-degree at most* ℓ, *and* $\mathrm{wdeg}_{1,k-1} Q(X, Y) < s(n - \tau)$. *Then* $(Y - f(X)) \mid Q(X, Y)$.

One can easily show that a polynomial $Q(X, Y)$ that fulfils the above conditions can be constructed whenever $E(s, \ell, \tau) > 0$, where

$$E(s, \ell, \tau) \triangleq (\ell + 1)s(n - \tau) - \binom{\ell+1}{2}(k - 1) - \binom{s+1}{2}n \tag{2}$$

is the difference between the maximal number of coefficients of $Q(X, Y)$, and the number of homogeneous linear equations on $Q(X, Y)$ specified by the interpolation constraint. This determines the maximal number of correctable errors, and one can show that satisfactory s and ℓ can always be chosen whenever $\tau < n - \sqrt{n(k - 1)}$ (for $n \to \infty$ see e.g. [7]).

Definition 2 (Permissible Triples) *An integer triple* $(s, \ell, \tau) \in (\mathbb{Z}_+)^3$ *is* permissible *if* $E(s, \ell, \tau) > 0$.

We define also the decoding radius-function $\tau(s, \ell)$ as the greatest integer such that $(s, \ell, \tau(s, \ell))$ is permissible.

It is easy to show that $E(s, \ell, \tau) > 0$ for $s > \ell$ implies $\tau < \lfloor \frac{n-k}{2} \rfloor$, which is half the minimum distance. Therefore, it never makes sense to consider $s > \ell$, and in the remainder we will always assume $s \le \ell$. Furthermore, we will also assume $s, \ell \in O(n^2)$ since this e.g. holds for any τ for the closed-form expressions in [7].

2.3 Module Reformulation of Guruswami–Sudan

Let $M_{s,\ell} \subset \mathbb{F}_q[X, Y]$ denote the space of all bivariate polynomials passing through the points $(\alpha_0, r'_0), \dots, (\alpha_{n-1}, r'_{n-1})$ with multiplicity s and with Y-degree at most ℓ. We are searching for an element of $M_{s,\ell}$ with low $(1, k - 1)$-weighted degree.

Following the ideas of Lee and O'Sullivan [8], we can first remark that $M_{s,\ell}$ is an $\mathbb{F}_q[X]$ module. Second, we can give an explicit basis for $M_{s,\ell}$. Define first two polynomials $G(X) = \prod_{i=0}^{n-1}(X - \alpha_i)$ as well as $R(X)$ as the Lagrange polynomial going through the points (α_i, r'_i) for $i = 0, \dots, n - 1$. Denote by $Q_{[t]}(X)$ the Y^t-coefficient of $Q(X, Y)$ when Q is regarded over $\mathbb{F}_q[X][Y]$.

Lemma 3 *Let* $Q(X, Y) \in M_{s,\ell}$. *Then* $G(X)^{s-t} \mid Q_{[t]}(X)$ *for* $t < s$.

Proof $Q(X, Y)$ interpolates the n points (α_i, r'_i) with multiplicity s, so for any i, $Q(X + \alpha_i, Y + r'_i) = \sum_{j=0}^{t} Q_{[j]}(X + \alpha_j)(Y + r'_j)^j$ has no monomials of total degree less than s. Multiplying out the $(Y + r'_j)^j$-terms, $Q_{[t]}(X + \alpha_j)Y^t$ will be the only term with Y-degree t. Therefore $Q_{[t]}(X + \alpha_j)$ can have no monomials of degree less than $s - t$, which implies $(X - \alpha_i) \mid Q_{[t]}(X)$. As this holds for any i, we proved the lemma. □

Theorem 4 *The module $M_{s,\ell}$ is generated as an $\mathbb{F}_q[X]$-module by the $\ell+1$ polynomials $P^{(i)}(X,Y) \in \mathbb{F}_q[X,Y]$ given by*

$$P^{(t)}(X,Y) = G(X)^{s-t}(Y - R(X))^t, \qquad \text{for } 0 \le t < s,$$
$$P^{(t)}(X,Y) = Y^{t-s}(Y - R(X))^s, \qquad \text{for } s \le t \le \ell.$$

Proof It is easy to see that each $P^{(t)}(X,Y) \in M_{s,\ell}$ since both $G(X)$ and $(Y - R(X))$ go through the n points (α_i, r_i') with multiplicity one, and that $G(X)$ and $(Y - R(X))$ divide $P^{(t)}(X,Y)$ with total power s for each t.

To see that any element of $M_{s,\ell}$ can be written as an $\mathbb{F}_q[X]$-combination of the $P^{(t)}(X,Y)$, let $Q(X,Y)$ be some element of $M_{s,\ell}$. Then the polynomial $Q^{(\ell-1)}(X,Y) = Q(X,Y) - Q_{[\ell]}P^{(\ell)}(X,Y)$ has Y-degree at most $\ell - 1$. Since both $Q(X,Y)$ and $P^{(\ell)}(X,Y)$ are in $M_{s,\ell}$, so must $Q^{(\ell-1)}(X,Y)$ be in $M_{s,\ell}$. Since $P^{(t)}(X,Y)$ has Y-degree t and $P^{(t)}_{[t]}(X) = 1$ for $t = \ell, \ell-1, \dots, s$, we can continue reducing this way until we reach a $Q^{(s-1)}(X,Y) \in M_{s,\ell}$ with Y-degree at most $s-1$. From then on, we have $P^{(t)}_{[t]}(X) = G(X)^{s-t}$, but by Lemma 3, we must also have $G(X) \mid Q^{(s-1)}_{[s-1]}(X)$, so we can also reduce by $P^{(s-1)}(X,Y)$. This can be continued with the remaining $P^{(t)}(X,Y)$, eventually reducing the remainder to 0. □

We can represent the basis of $M_{s,\ell}$ by the $(\ell+1) \times (\ell+1)$ matrix $\mathcal{A}_{s,\ell} = \|P^{(i)}_{[j]}(X,Y)\|_{i=0,j=0}^{\ell,\ell}$ over $\mathbb{F}_q[X]$. Any $\mathbb{F}_q[X]$-linear combination of rows of $\mathcal{A}_{s,\ell}$ thus corresponds to an element in $M_{s,\ell}$ by its tth term being the $\mathbb{F}_q[X]$-coefficient to Y^t. All other bases of $M_{s,\ell}$ can be similarly represented by matrices, and these will be unimodular equivalent to $\mathcal{A}_{s,\ell}$, i.e., they can be obtained by multiplying $\mathcal{A}_{s,\ell}$ on the left with an invertible matrix over $\mathbb{F}_q[X]$.

Extending the work of Lee and O'Sullivan [8], Beelen and Brander [2] gave a fast algorithm for computing a satisfactory $Q(X,Y)$: start with $\mathcal{A}_{s,\ell}$ as a basis of $M_{s,\ell}$ and compute a different, "minimal" basis of $M_{s,\ell}$ where an element of minimal $(1, k-1)$-weighted degree appears directly.[1]

In the following section, we give further details on how to compute such a basis, but our ultimate aims in Section 4 are different: we will use a minimal basis of $M_{s,\ell}$ to efficiently compute one for $M_{\hat{s},\hat{\ell}}$ for $\hat{s} \ge s$ and $\hat{\ell} > \ell$. This will allow an iterative refinement for increasing s and ℓ, where after each step we have such a minimal basis for $M_{s,\ell}$. We then exploit this added flexibility in our multi-trial algorithm.

3 Module Minimisation

Given a basis of $M_{s,\ell}$, e.g. $\mathcal{A}_{s,\ell}$, the module minimisation here refers to the process of obtaining a new basis, which is the smallest among all bases of $M_{s,\ell}$ in a precise sense. We will define this and connect various known properties of such matrices, and use this to more precisely bound the asymptotic complexity with which they can be computed by Alekhnovich's algorithm.

Definition 5 (Weak Popov Form [10]) *A matrix $\mathcal{V}$ over $\mathbb{F}_q[X]$ is in* weak Popov *form if an only if the leading position of each row is different.*

[1] Actually, in both [8, 2], a slight variant of $\mathcal{A}_{s,\ell}$ is used, but the difference is non-essential.

We are essentially interested in short vectors in a module, and the following lemma shows that the simple concept of weak Popov form will provide this. It is a paraphrasing of [1, Proposition 2.3] and we omit the proof.

Lemma 6 (Minimal Degree) *If a square matrix $\mathcal{V}$ over $\mathbb{F}_q[X]$ is in weak Popov form, then one of its rows has minimal degree of all vectors in the row space of $\mathcal{V}$.*

Denote now by $\mathcal{W}_\ell$ the diagonal $(\ell+1) \times (\ell+1)$ matrix over $\mathbb{F}_q[X]$:

$$\mathcal{W}_\ell \triangleq \operatorname{diag}\left(1, X^{k-1}, \ldots, X^{\ell(k-1)}\right). \tag{3}$$

Since we seek an element of minimal $(1, k-1)$-weighted degree, we also need the following corollary.

Corollary 7 (Minimal Weighted Degree) *Let $\mathcal{B} \in \mathbb{F}_q[X]^{(\ell+1)\times(\ell+1)}$ be the matrix representation of a basis of $M_{s,\ell}$. If $\mathcal{B}\mathcal{W}_\ell$ is in weak Popov form, then one of the rows of $\mathcal{B}$ corresponds to a polynomial in $M_{s,\ell}$ with minimal $(1, k-1)$-weighted degree.*

Proof Let $\widetilde{\mathcal{B}} = \mathcal{B}\mathcal{W}_\ell$. Now, $\widetilde{\mathcal{B}}$ will correspond to the basis of an $\mathbb{F}_q[X]$-module $\widetilde{M}$ isomorphic to $M_{s,\ell}$, where an element $Q(X,Y) \in M_{s,\ell}$ is mapped to $Q(X, X^{k-1}Y) \in \widetilde{M}$. By Lemma 6, the row of minimal degree in $\widetilde{\mathcal{B}}$ will correspond to an element of $\widetilde{M}$ with minimal X-degree. Therefore, the same row of $\mathcal{B}$ corresponds to an element of $M_{s,\ell}$ with minimal $(1, k-1)$-weighted degree. □

We introduce what will turn out to be a measure of how far a matrix is from being in weak Popov form.

Definition 8 (Orthogonality Defect [9]) *Let the* orthogonality defect *of a square matrix $\mathcal{V}$ over $\mathbb{F}_q[X]$ be defined as $\mathrm{D}(\mathcal{V}) \triangleq \deg \mathcal{V} - \deg\det \mathcal{V}$.*

Lemma 9 *If a square matrix $\mathcal{V}$ over $\mathbb{F}_q[X]$ is in weak Popov form then $\mathrm{D}(\mathcal{V}) = 0$.*

Proof Let $\mathbf{v}_0, \ldots, \mathbf{v}_{m-1}$ be the rows of $\mathcal{V} \in \mathbb{F}_q[X]^{m\times m}$ and $v_{i,0}, \ldots, v_{i,m-1}$ the elements of $\mathbf{v}_i$. In the alternating sum-expression for $\det \mathcal{V}$, the term $\prod_{i=0}^{m-1} \mathrm{LT}(\mathbf{v}_i)$ will occur since the leading positions of $\mathbf{v}_i$ are all different. Thus $\deg\det\mathcal{V} = \sum_{i=0}^{m-1} \deg \mathrm{LT}(\mathbf{v}_i) = \deg \mathcal{V}$ unless leading term cancellation occurs in the determinant expression. However, no other term in the determinant has this degree: regard some (unsigned) term in $\det \mathcal{V}$, say $t = \prod_{i=0}^{m-1} v_{i,\sigma(i)}$ for some permutation $\sigma \in S_m$. If not $\sigma(i) = \mathrm{LP}(\mathbf{v}_i)$ for all i, then there must be an i such that $\sigma(i) > \mathrm{LP}(\mathbf{v}_i)$ since $\sum_j \sigma(j)$ is the same for all $\sigma \in S_m$. Thus, $\deg v_{i,\sigma(i)} < \deg v_{i,\mathrm{LP}(\mathbf{v}_i)}$. As none of the other terms in t can have greater degree than their corresponding row's leading term, we get $\deg t < \sum_{i=0}^{m-1} \deg \mathrm{LT}(\mathbf{v}_i)$. Thus, $\mathrm{D}(\mathcal{V}) = 0$. However, the above also proves that the orthogonality defect is at least 0 for *any* matrix. Since any matrix unimodular equivalent to $\mathcal{V}$ has the same determinant, $\mathcal{V}$ must therefore have minimal row-degree among these matrices. □

Alekhnovich [1] gave a fast algorithm for transforming a matrix over $\mathbb{F}_q[X]$ to weak Popov form. For the special case of square matrices, a finer description of its asymptotic complexity can be reached in terms of the orthogonality defect, and this is essential for our decoder.

Lemma 10 (Alekhnovich's Row-Reducing Algorithm) *Alekhnovich's algorithm inputs a matrix $\mathcal{V} \in \mathbb{F}_q[X]^{m\times m}$ and outputs a unimodular equivalent matrix which is in weak Popov form. Let N be the greatest degree of a term in $\mathcal{V}$. If $N \in O(\mathrm{D}(\mathcal{V}))$ then the algorithm has asymptotic complexity:*

$$O\big(m^3\,\mathrm{D}(\mathcal{V})\log^2\mathrm{D}(\mathcal{V})\log\log\mathrm{D}(\mathcal{V})\big) \quad \textit{operations over } \mathbb{F}_q.$$

Proof The description of the algorithm as well as proof of its correctness can be found in [1]. We only prove the claim on the complexity. The method $R(\mathcal{V}, t)$ of [1] computes a unimodular matrix $\mathcal{U}$ such that $\deg(\mathcal{U}\mathcal{V}) \le \deg\mathcal{V} - t$ or $\mathcal{U}\mathcal{V}$ is in weak Popov form. According to [1, Lemma 2.10], the asymptotic complexity of this computation is in $O(m^3 t\log^2 t\log\log t)$. Due to Lemma 9, we can set $t = \mathrm{D}(\mathcal{V})$ to be sure that $\mathcal{U}\mathcal{V}$ is in weak Popov form. What remains is just to compute the product $\mathcal{U}\mathcal{V}$. Due to [1, Lemma 2.8], each entry in $\mathcal{U}$ can be represented as $p(X)X^d$ for some $d \in \mathbb{N}_0$ and $p(X) \in \mathbb{F}_q[X]$ of degree at most $2t$. If therefore $N \in O(\mathrm{D}(\mathcal{V}))$, the complexity of performing the matrix multiplication using the naive algorithm is $O(m^3\,\mathrm{D}(\mathcal{V}))$. □

4 Multi-Trial List Decoding

4.1 Basic Idea

Using the results of the preceding section, we show in Section 4.2 that given a basis of $M_{s,\ell}$ as a matrix $\mathcal{B}_{s,\ell}$ in weak Popov form, then we can write down a matrix $\mathcal{C}^{\mathrm{I}}_{s,\ell+1}$ which is a basis of $M_{s,\ell+1}$ and whose orthogonality defect is much lower than that of $\mathcal{A}_{s,\ell+1}$. This means that reducing $\mathcal{C}^{\mathrm{I}}_{s,\ell+1}$ to weak Popov form using Alekhnovich's algorithm is faster than reducing $\mathcal{A}_{s,\ell+1}$. We call this kind of refinement a "micro-step of type I". In Section 4.3, we similarly give a way to refine a basis of $M_{s,\ell}$ to one of $M_{s+1,\ell+1}$, and we call this a micro-step of type II.

If we first compute a basis in weak Popov form of $M_{1,1}$ using $\mathcal{A}_{1,1}$, we can perform a sequence of micro-steps of type I and II to compute a basis in weak Popov form of $M_{s,\ell}$ for any s, ℓ with $\ell \ge s$. After any step, having some intermediate $\hat{s} \le s$, $\hat{\ell} \le \ell$, we will thus have a basis of $M_{\hat{s},\hat{\ell}}$ in weak Popov form. By Corollary 7, we could extract from $\mathcal{B}_{\hat{s},\hat{\ell}}$ a $\hat{Q}(X,Y) \in M_{\hat{s},\hat{\ell}}$ with minimal $(1, k-1)$-weighted degree. Since it must satisfy the interpolation conditions of Theorem 1, and since the weighted degree is minimal among such polynomials, it must also satisfy the degree constraints for $\hat{\tau} = \tau(\hat{s}, \hat{\ell})$. By that theorem any codeword with distance at most $\hat{\tau}$ from $\mathbf{r}$ would then be represented by a root of $\hat{Q}(X,Y)$.

Algorithm 1 is a generalisation and formalisation of this method. For a given $\mathcal{GRS}(n,k)$ code, one chooses ultimate parameters (s, ℓ, τ) being a permissible triple with $s \le \ell$. One also chooses a list of micro-steps and chooses after which micro-steps to attempt decoding; these choices are represented by a list of $\mathsf{S}_1, \mathsf{S}_2$ and Root elements. This list must contain exactly $s - \ell$ S_1-elements of and $s - 1$ S_2-elements, as it begins by computing a basis for $M_{1,1}$ and will end with a basis for $M_{s,\ell}$. If there is a Root element in the list, the algorithm finds all codewords with distance at most $\hat{\tau} = \tau(\hat{s}, \hat{\ell})$ from $\mathbf{r}$; if this list is non-empty, the computation breaks and the list is returned.

The algorithm calls sub-functions which we explain informally: $\mathsf{MicroStep1}$ and $\mathsf{MicroStep2}$ will take $\hat{s}, \hat{\ell}$ and a basis in weak Popov form for $M_{\hat{s},\hat{\ell}}$ and return a basis in weak Popov form for $M_{\hat{s},\hat{\ell}+1}$ respectively $M_{\hat{s}+1,\hat{\ell}+1}$; more detailed descriptions for these are given in Subsections 4.2 and 4.3. $\mathsf{MinimalWeightedRow}$ finds a polynomial of minimal $(1, k-1)$-weighted degree in $\mathcal{M}_{\hat{s},\hat{\ell}}$ given a basis in weak Popov form (Corollary 7). Finally, $\mathsf{RootFinding}(Q, \tau)$ returns all Y-roots of $Q(X, Y)$ of degree less than k and whose corresponding codeword has distance at most τ from the received word $\mathbf{r}$.

Algorithm 1: Multi-Trial Guruswami–Sudan Decoding

Input: A $\mathcal{GRS}(n, k)$ code and the received vector $\mathbf{r} = (r_0, \ldots, r_{n-1})$
A permissible triple (s, ℓ, τ)
A list C with elements in $\{\mathsf{S}_1, \mathsf{S}_2, \mathsf{Root}\}$ with $s-1$ instances of S_2, $\ell - s$ instances of S_1

Preprocessing: Calculate $r'_i = r_i/w_i$ for all $i = 0, \ldots, n-1$
Construct $\mathcal{A}_{1,1}$, and compute $\mathcal{B}_{1,1}$ from $\mathcal{A}_{1,1}\mathcal{W}_1$ using Alekhnovich's algorithm
Initial parameters $(\hat{s}, \hat{\ell}) \leftarrow (1, 1)$

for *each* c *in* C **do**
    if $c = \mathsf{S}_1$ **then**
        $\mathcal{B}_{\hat{s},\hat{\ell}+1} \leftarrow \mathsf{MicroStep1}(\hat{s}, \hat{\ell}, \mathcal{B}_{\hat{s},\hat{\ell}})$
        $(\hat{s}, \hat{\ell}) \leftarrow (\hat{s}, \hat{\ell}+1)$
    if $c = \mathsf{S}_2$ **then**
        $\mathcal{B}_{\hat{s}+1,\hat{\ell}+1} \leftarrow \mathsf{MicroStep2}(\hat{s}, \hat{\ell}, \mathcal{B}_{\hat{s},\hat{\ell}})$
        $(\hat{s}, \hat{\ell}) \leftarrow (\hat{s}+1, \hat{\ell}+1)$
    if $c = \mathsf{Root}$ **then**
        $Q(X, Y) \leftarrow \mathsf{MinimalWeightedRow}(\mathcal{B}_{\hat{s},\hat{\ell}})$
        if $\mathsf{RootFinding}(Q(X, Y), \tau(\hat{s}, \hat{\ell})) \neq \emptyset$ **then**
            return this list

Algorithm 1 has a large amount of flexibility in the choice of the list C, but since we can only perform micro-steps of type I and II, there are choices of s and ℓ we can never reach, or some which we cannot reach if we first wish to reach an earlier s and ℓ. We can never reach $s > \ell$, but as mentioned in Section 2, such a choice never makes sense. It also seems to be the case that succession of sensibly chosen parameters can always be reached by micro-steps of type I and II. That is, if we first wish to attempt decoding at some radius τ_1 and thereafter continue to $\tau_2 > \tau_1$ in case of failure, the minimal possible s_1, ℓ_1 and s_2, ℓ_2 such that (s_1, ℓ_1, τ_1) respectively (s_2, ℓ_2, τ_2) are permissible will satisfy $0 \leq s_2 - s_1 \leq \ell_2 - \ell_1$. However, we have yet to formalise and prove such a statement.

In the following two subsections we explain the details of the micro-steps. In Section 4.4, we discuss the complexity of the method and how the choice of C influence this.

4.2 Micro-Step Type I: $(s, \ell) \mapsto (s, \ell + 1)$

Lemma 11 *If $B^{(0)}(X, Y), \ldots, B^{(\ell)}(X, Y)$ is a basis of $M_{s,\ell}$, then the following is a basis of $M_{s,\ell+1}$:*

$$B^{(0)}(X, Y), \ \ldots \ , \ B^{(\ell)}(X, Y), Y^{\ell-s+1}(Y - R(X))^s$$

Proof In the basis of $M_{s,\ell+1}$ given in Theorem 4, the first $\ell+1$ generators are the generators of $M_{s,\ell}$. Thus all of these can be described by any basis of $M_{s,\ell+1}$. The last remaining generator is exactly $Y^{\ell-s+1}(Y-R(X))^s$. $\square$

In particular, the above lemma holds for a basis of $M_{s,\ell+1}$ in weak Popov form, represented by a matrix $\mathcal{B}_{s,\ell}$. The following matrix thus represents a basis of $M_{s,\ell+1}$:

$$\mathcal{C}^{\mathrm{I}}_{s,\ell+1} = \left[\begin{array}{c|c} \mathcal{B}_{s,\ell} & \mathbf{0}^T \\ \hline 0 \ldots 0 \ (-R)^s \ \binom{s}{1}(-R)^{s-1} \ \ldots & 1 \end{array}\right]. \tag{4}$$

Lemma 12 $\mathrm{D}(\mathcal{C}^{\mathrm{I}}_{s,\ell+1}\mathcal{W}_{\ell+1}) = s(\deg R - k + 1) \le s(n-k)$.

Proof We calculate the two quantities $\det(\mathcal{C}^{\mathrm{I}}_{s,\ell+1}\mathcal{W}_{\ell+1})$ and $\deg(\mathcal{C}^{\mathrm{I}}_{s,\ell+1}\mathcal{W}_{\ell+1})$. It is easy to see that

$$\det(\mathcal{C}^{\mathrm{I}}_{s,\ell+1}\mathcal{W}_{\ell+1}) = \det \mathcal{B}_{s,\ell} \det \mathcal{W}_{\ell+1} = \det \mathcal{B}_{s,\ell} \det \mathcal{W}_{\ell} X^{(\ell+1)(k-1)}.$$

For the row-degree, it is clearly $\deg(\mathcal{B}_{s,\ell}\mathcal{W}_\ell)$ plus the row-degree of the last row. If and only if the received word is not a codeword then $\deg R \ge k$, then the leading term of the last row must be $(-R)^s X^{(\ell+1-s)(k-1)}$. Thus, we get

$$\begin{aligned} \mathrm{D}(\mathcal{C}^{\mathrm{I}}_{s,\ell+1}\mathcal{W}_{\ell+1}) &= \big(\deg(\mathcal{B}_{s,\ell}\mathcal{W}_\ell) + s\deg R + (\ell+1-s)(k-1)\big) \\ &\quad - \big(\deg\det(\mathcal{B}_{s,\ell}\mathcal{W}_\ell) + (\ell+1)(k-1)\big) \\ &= s(\deg R - k + 1), \end{aligned}$$

where the last step follows from Lemma 9 as $\mathcal{B}_{s,\ell}\mathcal{W}_\ell$ is in weak Popov form. $\square$

Corollary 13 *The complexity of* $\mathsf{MicroStep1}(s,\ell,\mathcal{B}_{s,\ell})$ *is* $O(\ell^3 sn \log^2 n \log\log n)$.

Proof Follows by Lemma 10. Since $s \in O(n^2)$ we can leave out the s in log-terms. $\square$

4.3 Micro-Step Type II: $(s,\ell) \mapsto (s+1,\ell+1)$

Lemma 14 *If* $B^{(0)}(X,Y),\ldots,B^{(\ell)}(X,Y)$ *is a basis of* $M_{s,\ell}$*, then the following is a basis of* $M_{s+1,\ell+1}$*:*

$$G^{s+1}(X),\ B^{(0)}(X,Y)(Y-R(X)),\ \ldots\ ,\ B^{(\ell)}(X,Y)(Y-R(X)).$$

Proof Denote by $P^{(0)}_{s,\ell}(X,Y),\ldots,P^{(\ell)}_{s,\ell}(X,Y)$ the basis of $M_{s,\ell}$ as given in Theorem 4, and by $P^{(0)}_{s+1,\ell+1}(X,Y),\ldots,P^{(\ell+1)}_{s+1,\ell+1}(X,Y)$ the basis of $M_{s+1,\ell+1}$. Then observe that for $t>0$, we have $P^{(t)}_{s+1,\ell+1} = P^{(t-1)}_{s,\ell}(Y-R(X))$. Since the $B^{(i)}(X,Y)$ form a basis of $M_{s,\ell}$, each $P^{(t)}_{s,\ell}$ is expressible as an $\mathbb{F}_q[X]$-combination of these, and thus for $t>0$, $P^{(t)}_{s+1,\ell+1}$ is expressible as an $\mathbb{F}_q[X]$-combination of the $B^{(i)}(X,Y)(Y-R(X))$. Remaining is then only $P^{(0)}_{s+1,\ell+1}(X,Y) = G^{s+1}(X)$. $\square$

As before, we can use the above with the basis $\mathcal{B}_{s,\ell}$ of $M_{s,\ell}$ in weak Popov form, found in the previous iteration of our algorithm. Remembering that multiplying by Y translates to shifting one column to the right in the matrix representation, the following matrix thus represents a basis of $M_{s+1,\ell+1}$:

$$\mathcal{C}^{\mathrm{II}}_{s+1,\ell+1} = \left[\begin{array}{c|c} G^{s+1} & \mathbf{0} \\ \hline \mathbf{0}^T & \mathbf{0} \end{array}\right] + \left[\begin{array}{c|c} 0 & \mathbf{0} \\ \hline \mathbf{0}^T & \mathcal{B}_{s,\ell} \end{array}\right] - R \cdot \left[\begin{array}{c|c} \mathbf{0} & 0 \\ \hline \mathcal{B}_{s,\ell} & \mathbf{0}^T \end{array}\right]. \tag{5}$$

Lemma 15 $\mathrm{D}(\mathcal{C}^{\mathrm{II}}_{s+1,\ell+1}\mathcal{W}_{\ell+1}) = (\ell+1)(\deg R - k + 1) \leq (\ell+1)(n-k)$.

Proof We compute $\deg(\mathcal{C}^{\mathrm{II}}_{s+1,\ell+1}\mathcal{W}_{\ell+1})$ and $\deg\det(\mathcal{C}^{\mathrm{II}}_{s+1,\ell+1}\mathcal{W}_{\ell+1})$. For the former, obviously the first row has degree $(s+1)n$. Let $\mathbf{b}_i$ denote the ith row of $\mathcal{B}_{s,\ell}$ and $\mathbf{b}'_i$ denote the ith row of $\mathcal{B}_{s,\ell}\mathcal{W}_\ell$. The $(i+1)$th row of $\mathcal{C}^{\mathrm{II}}_{s+1,\ell+1}\mathcal{W}_{\ell+1}$ has the form

$$\big[(0 \mid \mathbf{b}_i) - R(\mathbf{b}_i \mid 0)\big]\mathcal{W}_{\ell+1} = (0 \mid \mathbf{b}'_i)X^{k-1} - R(\mathbf{b}'_i \mid 0).$$

If and only if the received word is not a codeword, then $\deg R \geq k$. In this case, the leading term of $R\mathbf{b}'_i$ must have greater degree than any term in $X^{k-1}\mathbf{b}'_i$. Thus the degree of the above row is $\deg R + \deg \mathbf{b}'_i$. Summing up we get

$$\begin{aligned} \deg \mathcal{C}^{\mathrm{II}}_{s+1,\ell+1} &= (s+1)n + \sum_{i=0}^{\ell} \deg R + \deg \mathbf{b}'_i \\ &= (s+1)n + (\ell+1)\deg R + \deg(\mathcal{B}_{s,\ell}\mathcal{W}_\ell). \end{aligned}$$

For the determinant, observe that

$$\begin{aligned} \det(\mathcal{C}^{\mathrm{II}}_{s+1,\ell+1}\mathcal{W}_{\ell+1}) &= \det(\mathcal{C}^{\mathrm{II}}_{s+1,\ell+1})\det(\mathcal{W}_{\ell+1}) \\ &= G^{s+1}\det\widetilde{\mathcal{B}}\det\mathcal{W}_\ell X^{(\ell+1)(k-1)}, \end{aligned}$$

where $\widetilde{\mathcal{B}} = \mathcal{B}_{s,\ell} - R\left[\grave{\mathcal{B}}_{s,\ell} \;\middle|\; \mathbf{0}^T\right]$ and $\grave{\mathcal{B}}_{s,\ell}$ is all but the zeroth column of $\mathcal{B}_{s,\ell}$. This means $\widetilde{\mathcal{B}}$ can be obtained by starting from $\mathcal{B}_{s,\ell}$ and iteratively adding the $(j+1)$th column of $\mathcal{B}_{s,\ell}$ scaled by $R(X)$ to the jth column, with j starting from 0 up to $\ell-1$. Since each of these will add a scaled version of an existing column in the matrix, this does not change the determinant. Thus, $\det\widetilde{\mathcal{B}} = \det\mathcal{B}_{s,\ell}$. But then $\det\widetilde{\mathcal{B}}\det\mathcal{W}_\ell = \det(\mathcal{B}_{s,\ell}\mathcal{W}_\ell)$ and so $\deg(\det\widetilde{\mathcal{B}}\det\mathcal{W}_\ell) = \deg(\mathcal{B}_{s,\ell}\mathcal{W}_\ell)$ by Lemma 9 since $\mathcal{B}_{s,\ell}\mathcal{W}_\ell$ is in weak Popov form. Thus we get

$$\deg\det(\mathcal{C}^{\mathrm{II}}_{s+1,\ell+1}\mathcal{W}_{\ell+1}) = (s+1)n + \deg(\mathcal{B}_{s,\ell}\mathcal{W}_\ell) + (\ell+1)(k-1).$$

The lemma follows from the difference of the two calculated quantities. □

Corollary 16 *The complexity of* $\mathsf{MicroStep2}(s,\ell,\mathcal{B}_{s,\ell})$ *is* $O(\ell^4 n \log^2 n \log\log n)$.

4.4 Complexity Analysis

Using the estimates of the two preceding subsections, we can make a rather precise worst-case asymptotic complexity analysis of our multi-trial decoder. The average running time will depend on the exact choice of C but we will see that the worst-case complexity will not. First, it is necessary to know the complexity of performing a root-finding attempt.

Lemma 17 (Complexity of Root-Finding) *Given a polynomial* $Q(X,Y) \in \mathbb{F}_q[X][Y]$ *of* Y*-degree at most* ℓ *and* X*-degree at most* N*, there exists an algorithm to find all* $\mathbb{F}_q[X]$*-roots of complexity* $O\big(\ell^2 N \log^2 N \log\log N\big)$*, assuming* $\ell, q \in O(N)$*.*

Proof We employ the Roth–Ruckenstein [11] root-finding algorithm together with the divide-and-conquer speed-up by Alekhnovich [1]. The complexity analysis in [1] needs to be slightly improved to yield the above, but see [3] for easy amendments.

Theorem 18 (Complexity of Algorithm 1) *For a given* $\mathcal{GRS}(n,k)$ *code, as well as a given list of steps* C *for Algorithm 1 with ultimate parameters* (s,ℓ,τ)*, the algorithm has worst-case complexity* $O(\ell^4 sn \log^2 n \log\log n)$*, assuming* $q \in O(n)$*.*

Proof The worst-case complexity corresponds to the case that we do not break early but run through the entire list C. Precomputing $\mathcal{A}_{s,\ell}$ using Lagrangian interpolation can be performed in $O(n \log^2 n \log\log n)$, see e.g. [5, p. 235], and reducing to $\mathcal{B}_{s,\ell}$ is in the same complexity by Lemma 10.

Now, C must contain exactly $\ell - s$ S_1-elements and $s-1$ S_2-elements. The complexities given in Corollaries 13 and 16 for some intermediate $\hat{s}, \hat{\ell}$ can be relaxed to s and ℓ. Performing $O(\ell)$ micro-steps of type I and $O(s)$ of type II is therefore in $O(\ell^4 sn \log^2 n \log\log n)$.

It only remains to count the root-finding steps. Obviously, it never makes sense to have two Root after each other in C, so after removing such possible duplicates, there can be at most ℓ elements Root. When we perform root-finding for intermediate $\hat{s}, \hat{\ell}$, we do so on a polynomial in $M_{\hat{s},\hat{\ell}}$ of minimal weighted degree, and by the definition of $M_{\hat{s},\hat{\ell}}$ as well as Theorem 1, this weighted degree will be less than $\hat{s}(n - \hat{\tau}) < sn$. Thus we can apply Lemma 17 with $N = sn$. $\square$

The worst-case complexity of our algorithm is equal to the average-case complexity of the Beelen–Brander [2] list decoder. However, Theorem 18 shows that we can choose as many intermediate decoding attempts as we would like without changing the worst-case complexity. One could therefore choose to perform a decoding attempt just after computing $\mathcal{B}_{1,1}$ as well as every time the decoding radius has increased. The result would be a decoding algorithm finding all *closest* codewords within some ultimate radius τ. If one is working in a decoding model where such a list suffices, our algorithm will thus have much better average-case complexity since fewer errors occur much more frequently than many.

5 Conclusion

An iterative interpolation procedure for list decoding GRS codes based on Alekhnovich's module minimisation was proposed and shown to have the same worst-case complexity as Beelen and Brander's [2]. We showed how the target module used in Beelen–Brander can be minimised in a progressive manner, starting with a small module and systematically enlarging it, performing module minimisation in each step. The procedure takes advantage of a new, slightly more fine-grained complexity analysis of Alekhnovich's algorithm, which implies that each of the module refinement steps will run fast.

The main advantage of the algorithm is its granularity which makes it possible to perform fast multi-trial decoding: we attempt decoding for progressively larger decoding radii, and therefore find the list of codewords closest to the received. This is done without a penalty in the worst case but with an obvious benefit in the average case.

Acknowledgement

The authors thank Daniel Augot for fruitful discussions. This work has been supported by German Research Council Deutsche Forschungsgemeinschaft (DFG) under grant BO 867/22-1. Johan S. R. Nielsen also gratefully acknowledges the support from The Otto Mønsted Foundation and the Idella Foundation.

References

1. Alekhnovich, M.: Linear Diophantine Equations Over Polynomials and Soft Decoding of Reed–Solomon Codes. IEEE Transactions on Information Theory **51**(7) (2005)
2. Beelen, P., Brander, K.: Key equations for list decoding of Reed–Solomon codes and how to solve them. Journal of Symbolic Computation **45**(7), 773–786 (2010)
3. Beelen, P., Høholdt, T., Nielsen, J.S.R., Wu, Y.: On Rational-Interpolation Based List-Decoding and List-Decoding Binary Goppa Codes. arXiv preprint arXiv:1211.0122 (2012)
4. Bernstein, D.J.: List Decoding for Binary Goppa Codes. In: IWCC, pp. 62–80 (2011)
5. von zur Gathen, J., Gerhard, J.: Modern Computer Algebra. Cambridge Univ Press (2003)
6. Giorgi, P., Jeannerod, C., Villard, G.: On the Complexity of Polynomial Matrix Computations. In: Proceedings of International Symposium on Symbolic and Algebraic Computation '03, pp. 135–142. ACM (2003)
7. Guruswami, V., Sudan, M.: Improved Decoding of Reed–Solomon Codes and Algebraic Geometry Codes. IEEE Transactions on Information Theory **45**(6), 1757–1767 (1999)
8. Lee, K., O'Sullivan, M.E.: List Decoding of Reed–Solomon Codes from a Gröbner Basis Perspective. Journal of Symbolic Computation **43**(9), 645 – 658 (2008)
9. Lenstra, A.: Factoring Multivariate Polynomials over Finite Fields. Journal of Computer and System Sciences **30**(2), 235–248 (1985)
10. Mulders, T., Storjohann, A.: On lattice reduction for polynomial matrices. Journal of Symbolic Computation **35**(4), 377–401 (2003)
11. Roth, R., Ruckenstein, G.: Efficient Decoding of Reed–Solomon Codes Beyond Half the Minimum Distance. IEEE Transactions on Information Theory **46**(1), 246 –257 (2000)
12. Sudan, M.: Decoding of Reed–Solomon Codes beyond the Error-Correction Bound. Journal of Complexity **13**(1), 180–193 (1997)
13. Zeh, A., Gentner, C., Augot, D.: An Interpolation Procedure for List Decoding Reed–Solomon Codes Based on Generalized Key Equations. IEEE Transactions on Information Theory **57**(9), 5946–5959 (2011)