Des. Codes Cryptogr. (2017) 82:389–409
DOI 10.1007/s10623-016-0257-9

Row reduction applied to decoding of rank-metric and subspace codes

Sven Puchinger[1] · Johan Rosenkilde né Nielsen[2] · Wenhui Li[1] · Vladimir Sidorenko[3]

Received: 12 October 2015 / Revised: 23 June 2016 / Accepted: 13 July 2016 /
Published online: 2 August 2016

Abstract We show that decoding of ℓ-Interleaved Gabidulin codes, as well as list-ℓ decoding of Mahdavifar–Vardy (MV) codes can be performed by row reducing skew polynomial matrices. Inspired by row reduction of $\mathbb{F}[x]$ matrices, we develop a general and flexible approach of transforming matrices over skew polynomial rings into a certain reduced form. We apply this to solve generalised shift register problems over skew polynomial rings which occur in decoding ℓ-Interleaved Gabidulin codes. We obtain an algorithm with complexity $O(\ell\mu^2)$ where μ measures the size of the input problem and is proportional to the code length n in the case of decoding. Further, we show how to perform the interpolation step of list-ℓ-decoding MV codes in complexity $O(\ell n^2)$, where n is the number of interpolation constraints.

Keywords Skew polynomials · Row reduction · Module minimisation · Gabidulin codes · Shift register synthesis · Mahdavifar–Vardy codes

This is one of several papers published in *Designs, Codes and Cryptography* comprising the "Special Issue on Coding and Cryptography".

Vladimir Sidorenko is on leave from Institute of Information Transmission Problems (IITP), Russian Academy of Sciences.

✉ Johan Rosenkilde né Nielsen
jsrn@jsrn.dk

Sven Puchinger
sven.puchinger@uni-ulm.de

Wenhui Li
wenhui.li@uni-ulm.de

Vladimir Sidorenko
vladimir.sidorenko@tum.de

[1] Institute of Communications Engineering, Ulm University, Ulm, Germany

[2] Department of Applied Mathematics and Computer Science, Technical University of Denmark, Kongens Lyngby, Denmark

[3] Institute for Communications Engineering, TU München, Munich, Germany

Mathematics Subject Classification 12Y05 · 12E15 · 11T71

1 Introduction

Numerous recent publications have unified the core of various decoding algorithms for Reed–Solomon (RS) and Hermitian codes using row reduction of certain $\mathbb{F}[x]$-module bases. First for the Guruswami–Sudan list decoder [2,9,21], then for Power decoding [32,33] and also either type of decoder for Hermitian codes [34]. By factoring out coding theory from the core problem, we enable the immediate use of sophisticated algorithms developed by the computer algebra community such as [16,50].

The goal of this paper is to explore the row reduction description over skew polynomial rings, with a main application for decoding rank-metric and subspace codes. Concretely, we prove that Interleaved Gabidulin and Mahdavifar–Vardy (MV) codes can be decoded by transforming a module basis into weak Popov form, which can be obtained by a skew-analogue of the elegantly simple Mulders–Storjohann algorithm [31]. By exploiting the structure of the module bases arising from the decoding problems, we refine the algorithm to obtain improved complexities. These match the best known algorithms for these applications but solve more general problems, and it demonstrates that the row reduction methodology is both flexible and fast for skew polynomial rings. Building on this paper, Ref. [40] proposes an algorithm which improves upon the best known complexity for decoding Interleaved Gabidulin codes.

Section 1.1 summarizes related work. We set basic notation in Sect. 2. Section 3 shows how to solve the mentioned decoding problems using row reduction and states the final complexity results which are proven in the subsequent sections. We describe row reduction of skew polynomial matrices in Sect. 4. Section 5 presents faster row reduction algorithms for certain input matrices, with applications to the decoding problems.

This work was partly presented at the International Workshop on Coding and Cryptography 2015 [24]. Compared to this previous work, we added the decoding of MV codes using the row reduction approach.[1] It spurred a new refinement of the Mulders–Storjohann, in Sect. 5.2, which could be of wider interest.

1.1 Related work

In this paper we consider skew polynomial rings over finite fields without derivations [38] (see Sect. 2.1 for this restricted definition of skew polynomials). This is the most relevant case for coding theory, partly because they are easier to compute with, though non-zero derivations have been used in some constructions [8]. All of the row reduction algorithms in this paper work for skew polynomial rings with non-zero derivation, but the complexity would be worse. The algorithms also apply to skew polynomial rings over any base ring, e.g. $\mathbb{F}(z)$ or a number field, but due to coefficient growth in such settings, their bit-complexity would have to be analysed.

A skew polynomial ring over a finite field without derivation is isomorphic to a ring of *linearised polynomials* under a trivial isomorphism, and the rings' evaluation maps agree. Our algorithms could be phrased straightforwardly to work on modules over linearised polynomials. Much literature on Gabidulin codes uses the language of linearised polynomials.

[1] We opted for using the term "row reduction" rather than "module minimisation", as we used in [24], since the former is more common in the literature.

Skew polynomial rings are instances of Ore rings, and some previous work on computing matrix normal forms over Ore rings can be found in [1,6]. The focus there is when the base ring is $\mathbb{Q}$ or $\mathbb{F}(z)$, where coefficient growth is a main concern, e.g. for modelling systems of partial differential equations. These algorithms are slower than ours when the base ring is a finite field. Reference [29] considers a setting more similar to ours, but obtains a different algorithm and slightly worse complexity.

Gabidulin codes [11,15,41] are maximum rank distance codes over finite fields; they are the rank-metric analogue of RS codes. An Interleaved Gabidulin code is a direct sum of several Gabidulin codes, similar to Interleaved RS codes. In a synchronised error model they allow an average-case error-correction capability far beyond half the minimum rank distance [25]. Decoding of Interleaved Gabidulin codes is often formulated as solving a simultaneous "Key Equation" [43]. Over $\mathbb{F}[x]$ this computational problem is also known as a multi-sequence shift-register synthesis [14,45], simultaneous Padé approximation [4], or vector rational function reconstruction [36]. This problem also has further generalisations, some of which have found applications in decoding of algebraic codes, e.g. [34,42,49]. In the computer algebra community, Padé approximations have been studied in a wide generality, e.g. [5]; to the best of our knowledge, analogous generalisations over skew polynomial rings have yet to see any applications.

Lately, there has been an interest in Gabidulin codes over number fields, with applications to space-time codes and low-rank matrix recovery [3]. Their decoding can also be reduced to a shift-register type problem [30], which could be solved using the algorithms in this paper (though again, one should analyse the bit-complexity).

MV codes [26,28] are subspace codes whose main interest lies in their property of being list-decodable beyond half the minimum distance. Their rate unfortunately tends to zero for increasing code lengths. In [27], Mahdavifar and Vardy presented a refined construction which can be decoded "with multiplicities" allowing a better decoding radius and rate; it is future work to adapt our algorithm to this case. The decoding of MV codes is is heavily inspired by the Guruswami–Sudan algorithm for RS codes [17], and our row reduction approach in Sect. 3.2 is similarly inspired by fast module-based algorithms for realising the Guruswami–Sudan [7,21].

Another family of rank-metric codes which can be decoded beyond half the minimum distance are Guruswami–Xing codes [19]. These can be seen simply as heavily punctured Gabidulin codes, and their decoding as a *virtual interleaving* of multiple Gabidulin codes. This leads to a decoder based on solving a simultaneous shift-register equation, where our algorithms apply. Guruswami–Wang codes [18] are Guruswami–Xing codes with a restricted message set, so the same decoding algorithm applies.

Over $\mathbb{F}[x]$, row reduction, and the related concept of order bases, have been widely studied and sophisticated algorithms have emerged, e.g. [2,16,50]. As a follow-up to this work, a skew-analogue of the algorithm in [2] was proposed in [40].

2 Preliminaries

2.1 Skew polynomials

Let $\mathbb{F}$ be a finite field and θ an $\mathbb{F}$-automorphism. Denote by $\mathcal{R} = \mathbb{F}[x; \theta]$ the non-commutative ring of *skew polynomials* over $\mathbb{F}$ (with zero derivation): elements of $\mathcal{R}$ are of the form $\sum_i a_i x^i$

with $a_i \in \mathbb{F}$, addition is as usual, while multiplication is defined by $xa = \theta(a)x$ for all $a \in \mathbb{F}$. When we say "polynomial", we will mean elements of $\mathcal{R}$. The definition of the degree of a polynomial is the same as for ordinary polynomials. See [38] for more details.

The evaluation map of $a \in \mathcal{R}$ is given as:

$$\begin{aligned} a(\cdot) := \mathrm{ev}_a(\cdot) \, : \, \mathbb{F} \to \mathbb{F} \\ \alpha \mapsto \textstyle\sum_i a_i \theta^i(\alpha). \end{aligned}$$

This is a group homomorphism on $(\mathbb{F}, +)$, and it is a linear map over the fixed field of θ. Furthermore, for two $a, b \in \mathcal{R}$ we have $\mathrm{ev}_{ab} = \mathrm{ev}_a \circ \mathrm{ev}_b$. This is sometimes known as operator evaluation, e.g. [8].

If $\mathbb{F}_q$ is the field fixed by θ for some prime power q, then $\mathbb{F} = \mathbb{F}_{q^s}$, $s \in \mathbb{Z}_{>0}$, and $\theta(a) = a^{q^i}$ for some $0 \le i < s$, i.e. a power of the Frobenius automorphism of $\mathbb{F}_{q^s} / \mathbb{F}_q$.

Definition 1 For $a, b, c \in \mathcal{R}$, we write $a \equiv b \mod c$ (*right modulo operation*) if there exists $d \in \mathcal{R}$ such that $a = b + dc$

In complexity estimates we count the total number of the following operations: $+, -, \cdot, /$ and θ^i for any $i \in \mathbb{Z}_{>0}$. For computing θ^i the assumption is that the Frobenius automorphism can be done efficiently in $\mathbb{F}_{q^s}$; this is reasonable since we can represent $\mathbb{F}_{q^s}$-elements using a normal basis over $\mathbb{F}_q$ (cf. [47, Sect. 2.1.2]): in this case, a^q for $a \in \mathbb{F}_{q^s}$ is simply the cyclic shift of a represented as an $\mathbb{F}_q$-vector over the normal basis.

2.2 Skew polynomial matrices

Free modules and matrices over $\mathcal{R}$ behave quite similarly to the $\mathbb{F}[x]$ case, keeping non-commutativity in mind:

- Any left sub-module $\mathcal{V}$ of $\mathcal{R}^m$ is free and admits a basis of at most m elements. Any two bases of $\mathcal{V}$ have the same number of elements.
- The rank of a matrix M over $\mathcal{R}$ is defined as the number of elements in any basis of the left $\mathcal{R}$-row space of M. The rows of two such matrices $M, M' \in \mathcal{R}^{n \times m}$ generate the same left module if and only if there exists a $U \in \mathrm{GL}_n(\mathcal{R})$ such that $M = UM'$, where $\mathrm{GL}_n(\mathcal{R})$ denotes the set of invertible $n \times n$ matrices over $\mathcal{R}$.

These properties follow principally from $\mathcal{R}$ being an Ore ring and therefore left Euclidean, hence left Principal Ideal Domain (PID), hence left Noetherian.[2] Moreover, $\mathcal{R}$ has a unique left skew field[3] of fractions $\mathcal{Q}$ from which it inherits its linear algebra properties. See e.g. [10, 13] for more details. In this paper we exclusively use the left module structure of $\mathcal{R}$, and we will often omit the "left" denotation.

We introduce the following notation for vectors and matrices over $\mathcal{R}$: Matrices are denoted by capital letters (e.g. V). The ith row of V is denoted by $\boldsymbol{v}_i$, the jth element of a vector $\boldsymbol{v}$ is v_j and $v_{i,j}$ is the (i, j)th entry of a matrix V. Indices start at 0.

- The *degree of a vector* $\boldsymbol{v}$ is $\deg \boldsymbol{v} := \max_i\{\deg v_i\}$ (and $\deg \boldsymbol{0} = -\infty$) and the *degree of a matrix* V is $\deg V := \sum_i\{\deg \boldsymbol{v}_i\}$.
- The *max-degree* of V is $\operatorname{maxdeg} V := \max_i\{\deg \boldsymbol{v}_i\} = \max_{i,j}\{\deg v_{i,j}\}$.
- The *leading position* of a non-zero vector $\boldsymbol{v}$ is $\mathrm{LP}(\boldsymbol{v}) := \max\{i \, : \, \deg v_i = \deg \boldsymbol{v}\}$, i.e. the *rightmost* position having maximal degree in the vector. Furthermore, we define the *leading term* $\mathrm{LT}(\boldsymbol{v}) := v_{\mathrm{LP}(\boldsymbol{v})}$ and $\mathrm{LC}(\boldsymbol{v})$ is the leading coefficient of $\mathrm{LT}(\boldsymbol{v})$.

[2] $\mathcal{R}$ is also right Euclidean, a right PID and right Noetherian, but we will only need its left module structure.

[3] Skew fields are sometimes known as "division rings".

2.3 The weak Popov form

Definition 2 A matrix V over $\mathcal{R}$ is in *weak Popov form* if the leading positions of all its non-zero rows are different.

The following lemma describes that the rows of a matrix in weak Popov form are minimal in a certain way. Its proof is exactly the same as for $\mathbb{F}[x]$ modules and is therefore omitted, see e.g. [32].

Lemma 1 *Let V be a matrix in weak Popov form, and let $\mathcal{V}$ be the $\mathcal{R}$-module generated by its rows. Then the non-zero rows of V are a basis of $\mathcal{V}$ and every $\boldsymbol{u} \in \mathcal{V}$ satisfies* $\deg \boldsymbol{u} \geq \deg \boldsymbol{v}$, *where $\boldsymbol{v}$ is the row of V with* $\mathrm{LP}(\boldsymbol{v}) = \mathrm{LP}(\boldsymbol{u})$.

We will need to "shift" the relative importance of some columns compared to others. Given a "shift vector" $\boldsymbol{w} = (w_0, \dots, w_\ell) \in \mathbb{Z}_{\geq 0}^{\ell+1}$, define the mapping

$$\Phi_{\boldsymbol{w}} : \mathcal{R}^{\ell+1} \to \mathcal{R}^{\ell+1},\ \boldsymbol{u} = (u_0, \dots, u_\ell) \mapsto \left(u_0 x^{w_0}, \dots, u_\ell x^{w_\ell}\right).$$

It is easy to compute the inverse of $\Phi_{\boldsymbol{w}}$ for any vector in $\Phi_{\boldsymbol{w}}(\mathcal{R}^{\ell+1})$. Note that since the monomials x^{w_i} are multiplied from the right, applying $\Phi_{\boldsymbol{w}}$ will only *shift* the entry polynomials, and not modify the coefficients. We can extend $\Phi_{\boldsymbol{w}}$ to $\mathcal{R}$-matrices by applying it row-wise.

Definition 3 For any $\boldsymbol{w} = (w_0, \dots, w_\ell) \in \mathbb{Z}_{\geq 0}^{\ell+1}$, a matrix $V \in \mathcal{R}^{\cdot \times (\ell+1)}$ is in $\boldsymbol{w}$*-shifted weak Popov* form if $\Phi_{\boldsymbol{w}}(V)$ is in weak Popov form.

Given some matrix V over $\mathcal{R}$, "transforming V into ($\boldsymbol{w}$-shifted) weak Popov form" means to find some W generating the same row space as V and such that W is in ($\boldsymbol{w}$-shifted) weak Popov form. We will see in Sect. 4.1 that such W always exist.

Throughout this paper, by "row reduced" we mean "in weak Popov form".[4] Similarly, "row reduction" means "transforming into weak Popov form".

3 Decoding problems in rank-metric and subspace codes

3.1 Interlaved Gabidulin codes: multi-sequence shift registers

It is classical to decode errors in a Gabidulin code by solving a syndrome-based "Key Equation": that is, a shift-register synthesis problem over $\mathcal{R}$, see e.g. [15]. An Interleaved Gabidulin code is a direct sum of several Gabidulin codes [25], and error-decoding can be formulated as a shift-register synthesis of several sequences simultaneously. A slightly more general notion of shift-register synthesis allows formulating the decoder using the "Gao Key Equation" [47]. Another generalisation accommodates error-and-erasure decoding of some Gabidulin resp. Interleaved Gabidulin codes [23,47].

All these approaches are instances of the following "Multi-Sequence generalised Linear Skew-Feedback Shift Register" (MgLSSR) synthesis problem:

[4] There is a precise notion of "row reduced" [20, p. 384] for $\mathbb{F}[x]$ matrices. Weak Popov form implies being row reduced, but we will not formally define row reduced in this paper.

Problem 1 (*MgLSSR*) Given skew polynomials $s_i, g_i \in \mathcal{R}$ and non-negative integers $\gamma_i \in \mathbb{Z}_{\geq 0}$ for $i = 1, \ldots, \ell$, find skew polynomials $\lambda, \omega_1, \ldots, \omega_\ell \in \mathcal{R}$, with λ of minimal degree such that the following holds:

$$\lambda s_i \equiv \omega_i \mod g_i$$
$$\deg \lambda + \gamma_0 > \deg \omega_i + \gamma_i$$

We show how to solve this problem by row reduction of a particular module basis. The approach is analogous to how the $\mathbb{F}[x]$-version of the problem is handled by Rosenkilde in [32], with only a few technical differences due to the non-commutativity of $\mathcal{R}$.

In the sequel we consider a particular instance of Problem 1, so $\mathcal{R}, \ell \in \mathbb{Z}_{>0}$, and $s_i, g_i \in \mathcal{R}$, $\gamma_i \in \mathbb{Z}_{\geq 0}$ for $i = 1, \ldots, \ell$ are arbitrary but fixed. We assume $\deg s_i \leq \deg g_i$ for all i since taking $s_i := (s_i \mod g_i)$ yields the same solutions to Problem 1.

Denote by $\mathcal{M}$ the set of all vectors $\boldsymbol{v} \in \mathcal{R}^{\ell+1}$ satisfying the congruence relation, i.e.,

$$\mathcal{M} := \left\{(\lambda, \omega_1, \ldots, \omega_\ell) \in \mathcal{R}^{\ell+1} \;\middle|\; \lambda s_i \equiv \omega_i \mod g_i \; \forall i = 1, \ldots, \ell\right\}. \tag{1}$$

Lemma 2 *Consider an instance of Problem 1 and $\mathcal{M}$ as in (1). $\mathcal{M}$ with component-wise addition and left multiplication by elements of $\mathcal{R}$ forms a free left module over $\mathcal{R}$. The rows of M form a basis of $\mathcal{M}$, where*

$$M = \begin{pmatrix} 1 & s_1 & s_2 & \ldots & s_\ell \\ 0 & g_1 & 0 & \ldots & 0 \\ 0 & 0 & g_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & g_\ell \end{pmatrix}. \tag{2}$$

Proof Since $\mathcal{M} \subseteq \mathcal{R}^{\ell+1}$, the first half of the statement follows easily since $\mathcal{M}$ is clearly closed under addition and left $\mathcal{R}$ multiplication. M is a basis of $\mathcal{M}$ by arguments analogous to the $\mathbb{F}[x]$ case, cf. [32, Lemma 1]. □

Lemma 2 gives a simple description of all solutions of the congruence requirement of Problem 1 in the form of the row space of an explicit matrix M. The following theorem implies that computing the weak Popov form of M is enough to solve Problem 1. The proof is similar to the $\mathbb{F}[x]$-case but since there is no convenient reference for it, we give it here for the $\mathcal{R}$ case. The entire strategy is formalised in Algorithm 1.

Theorem 1 *Consider an instance of Problem 1, and $\mathcal{M}$ as in (1). Let $\boldsymbol{w} = (\gamma_0, \ldots, \gamma_\ell) \in \mathbb{Z}_{\geq 0}^{\ell+1}$. If V is a basis of $\mathcal{M}$ in $\boldsymbol{w}$-shifted weak Popov form, the row $\boldsymbol{v}$ of V with* $\mathrm{LP}(\Phi_{\boldsymbol{w}}(\boldsymbol{v})) = 0$ *is a solution to Problem 1.*

Proof By Lemma 2 the row $\boldsymbol{v}$ satisfies the congruence requirement of Problem 1. For the degree restriction of Problem 1, note that any $\boldsymbol{u} \in \mathcal{M}$ satisfies this restriction if and only if $\mathrm{LP}(\Phi_{\boldsymbol{w}}(\boldsymbol{u})) = 0$, since $\deg u_i + \gamma_i = \deg(\Phi_{\boldsymbol{w}}(\boldsymbol{u})_i)$. Furthermore, if this is the case, then $\deg(\Phi_{\boldsymbol{w}}(\boldsymbol{u})) = \deg u_0 + \gamma_0$. Thus, not only must $\boldsymbol{v}$ satisfy the degree restriction, but by Lemma 1, v_0 also has minimal possible degree. □

Algorithm 1 Solve Problem 1 by row reduction

```
Input: Instance of Problem 1.
Output: Solution $\boldsymbol{v} = (\lambda, \omega_1, \ldots, \omega_\ell)$ of Problem 1.
1 Set up $M$ as in 2.
2 Compute $V$ as a $\boldsymbol{w}$-shifted weak Popov form of $M$.
3 return the row $\boldsymbol{v}$ of $V$ having $\mathrm{LP}(\Phi_{\boldsymbol{w}}(\boldsymbol{v})) = 0$.
```

The complexity of Algorithm 1 is determined by Line 2. Therefore, in Sects. 4 and 5.1 we analyse how and in which complexity we can row-reduce $\mathcal{R}$-matrices. In particular, we prove the following statement, where $\mu := \max_i\{\gamma_i + \deg g_i\}$.

Theorem 2 *Algorithm 1 has complexity* $\begin{cases} O(\ell\mu^2), & \text{if } g_i = x^{t_i} + c_i,\ t_i \in \mathbb{Z}_{>0},\ c_i \in \mathbb{F}\ \forall i, \\ O(\ell^2\mu^2), & \text{otherwise.}\end{cases}$

Proof The first case follows from Theorem 8 in Sect. 5.1, using Algorithm 4 for the row reduction step. For general g_i's, the result of Example 2 in Sect. 4 holds, which estimates the complexity of Algorithm 3 for a shift-register input. □

The above theorem applies well to decoding Gabidulin and Interleaved Gabidulin codes since the g_i are often in the restricted form: specifically, g_i is a power of x in syndrome Key Equations, while $g_i = x^n - 1$ in Gao Key Equation whenever $n \mid s$. We therefore achieve the same complexity as [44] but in a wider setting.

3.2 Decoding Mahdavifar–Vardy codes

MV codes [26,28] are subspace codes constructed by evaluating powers of skew polynomials at certain points. We will describe how one can use row reduction to carry out the most computationally intensive step of the MV decoding algorithm given in [28], the Interpolation step. In this section, $\mathcal{R} = \mathbb{F}_{q^s}[x;\theta]$ where θ is some power of the Frobenius automorphism of $\mathbb{F}_{q^s}/\mathbb{F}_q$.

Problem 2 (*Interpolation Step of MV decoding*) Let $\ell, k, s, n \in \mathbb{Z}_{>0}$ be such that $\binom{\ell+1}{2}(k-1) < n \le s$. Given $(x_i, y_{i,1}, \ldots, y_{i,\ell}) \in \mathbb{F}_{q^s}^{\ell+1}$ for $i = 1, \ldots, n$, where the x_i are linearly independent over $\mathbb{F}_q$, find a non-zero $\boldsymbol{Q} \in \mathcal{R}^{\ell+1}$ satisfying:

$$Q_0(x_i) + \sum_{t=1}^{\ell} Q_t(y_{i,t}) = 0 \qquad i = 1, \ldots, n, \tag{3}$$

$$\deg Q_t < \chi - t(k-1) \quad t = 0, \ldots, \ell, \tag{4}$$

where χ is given by

$$\chi = \left\lceil \frac{n+1}{\ell+1} + \frac{1}{2}\ell(k-1) \right\rceil.$$

The problem can be solved by a large linear system of equations whose dimensions reveals that a solution always exists [28, Lemma 8]. Note that the requirement $n > \binom{\ell+1}{2}(k-1)$ ensures that all the degree bounds (4) are non-negative.

Let $\mathcal{M}$ be the set of all $\boldsymbol{Q}$ that satisfy (3) though not necessarily (4):

$$\mathcal{M} = \left\{ \boldsymbol{Q} \in \mathcal{R}^{\ell+1} \;\middle|\; Q_0(x_i) + \textstyle\sum_{t=1}^{\ell} Q_t(y_{i,t}) = 0 \quad i = 1, \ldots, n \right\} \tag{5}$$

Lemma 3 *Consider an instance of Problem 2. Then $\mathcal{M}$ of (5) is a left $\mathcal{R}$-module.*

Proof $\mathcal{M}$ is closed under addition since $a(\alpha) + b(\alpha) = (a+b)(\alpha)$ for all $a, b \in \mathcal{R}$ and $\alpha \in \mathbb{F}_{q^s}$. Let $f \in \mathcal{R}$, $\boldsymbol{Q} = (Q_0, Q_1, \ldots, Q_\ell) \in \mathcal{M}$. Then $f \cdot \boldsymbol{Q}$ satisfies (3) since

$$(f \cdot Q_0)(x) + \sum_{i=1}^{\ell} (f \cdot Q_i)(y_i) = f\left(Q_0(x) + \sum_{i=1}^{\ell} Q_i(y_i) \right) = f(0) = 0\,.$$

□

For explicitly describing a basis of $\mathcal{M}$, we need a few well-known technical elements:

Definition 4 Given $a_1, \dots, a_m \in \mathbb{F}_{q^s}$ which are linearly independent over $\mathbb{F}_q$, the *annihilator polynomial* of the a_i is the monic non-zero $\mathcal{A} \in \mathcal{R}$ of minimal degree such that $\mathcal{A}(a_i) = 0$ for all i.

It is easy to show that the annihilator polynomial is well-defined and that $\deg \mathcal{A} = m$, see e.g. [37]. The existence of annihilator polynomials easily leads to the following analogue of Lagrange interpolation:

Lemma 4 (Interpolation polynomial) *Given any $a_1, \dots, a_m \in \mathbb{F}_{q^s}$ which are linearly independent over $\mathbb{F}_q$, and arbitrary $b_1, \dots, b_m \in \mathbb{F}_{q^s}$, there exists a unique $R \in \mathcal{R}$ of degree at most $m - 1$ such that $R(a_i) = b_i$ for all $i = 1, \dots, m$.*

Lemma 5 *Consider an instance of Problem 2 and let $\mathcal{M}$ be as in (5). Denote by G the annihilator polynomial of the x_i, $i = 1, \dots, n$, and let $R_t \in \mathcal{R}$, $t = 1, \dots, \ell$ be the interpolation polynomial with $R_t(x_i) = y_{i,t}$ for $i = 1, \dots, n$. The rows of M form a basis of $\mathcal{M}$:*

$$M = \begin{pmatrix} \boldsymbol{m}_0 \\ \boldsymbol{m}_1 \\ \boldsymbol{m}_2 \\ \vdots \\ \boldsymbol{m}_\ell \end{pmatrix} = \begin{pmatrix} G & 0 & 0 & \dots & 0 \\ -R_1 & 1 & 0 & \dots & 0 \\ -R_2 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -R_\ell & 0 & 0 & \dots & 1 \end{pmatrix} \tag{6}$$

Proof "$\subseteq$": We should show that each $\boldsymbol{m}_j$ all "vanish" at the points $(x_i, y_{i,1}, \dots, y_{i,\ell})$. Consider such a point; we have two cases:

$$\boldsymbol{m}_0 : G(x_i) = 0$$
$$\boldsymbol{m}_t : 1(y_{i,t}) - R_t(x_i) = y_{i,t} - R_t(x_i) = 0\,, t = 1, \dots, \ell$$

"$\supseteq$": Consider some $\boldsymbol{Q} = (Q_0, \dots, Q_\ell) \in \mathcal{M}$. Then we can write

$$\begin{aligned} \boldsymbol{v}_\ell &:= \boldsymbol{Q} \\ \boldsymbol{v}_{\ell-1} &:= \boldsymbol{v}_\ell - v_{\ell,\ell} \cdot \boldsymbol{m}_\ell = (v_{\ell-1,0}, \dots, v_{\ell-1,\ell-1}, 0) \\ \boldsymbol{v}_{\ell-2} &:= \boldsymbol{v}_{\ell-1} - v_{\ell-1,\ell-1} \cdot \boldsymbol{m}_{\ell-1} = (v_{\ell-2,0}, \dots, v_{\ell-2,\ell-2}, 0, 0) \\ &\vdots \\ \boldsymbol{v}_0 &:= \boldsymbol{v}_1 - v_{1,1} \cdot \boldsymbol{m}_1 = (v_{0,0}, 0, \dots, 0). \end{aligned}$$

Since $\boldsymbol{v}_\ell \in \mathcal{M}$, and each $\boldsymbol{m}_t \in \mathcal{M}$, we conclude that all the $\boldsymbol{v}_t \in \mathcal{M}$ and in particular $\boldsymbol{v}_0 \in \mathcal{M}$. Thus for any i we must have $v_{0,0}(x_i) = 0$. This means G must right-divide $v_{0,0}$: for otherwise, the division would yield a non-zero remainder $B \in \mathcal{R}$ with $\deg B < \deg G$ but still having $B(x_i) = 0$, contradicting the minimality of G.

Summarily, $\boldsymbol{v}_0 = f \cdot \boldsymbol{m}_0$ for some $f \in \mathcal{R}$, and hence $\boldsymbol{Q} = \boldsymbol{v}_\ell$ is an $\mathcal{R}$-linear combination of the rows of M. □

To complete the interpolation step, we need to find an element of $\mathcal{M}$ whose components satisfy the degree constraints (4).

Theorem 3 *Consider an instance of Problem 2, and let $\mathcal{M}$ be as in (5). Let $\boldsymbol{w} = (0, (k - 1), \dots, \ell(k - 1))$, and V be a basis of $\mathcal{M}$ in $\boldsymbol{w}$-shifted weak Popov form. If $\boldsymbol{v}$ is a row of V with minimal $\boldsymbol{w}$-shifted degree,* $\deg \Phi_{\boldsymbol{w}}(\boldsymbol{v})$*, then $\boldsymbol{v}$ is a solution to Problem 2.*

Proof Any row of V satisfies (3) because it is in $\mathcal{M}$. As previously remarked, there exists some solution $\boldsymbol{Q} = (Q_0, Q_1, \ldots, Q_\ell) \in \mathcal{M}$ satisfying the degree conditions (4). By the choice of $\boldsymbol{v}$ and by Lemma 1 on page 5, then $\deg \Phi_{\boldsymbol{w}}(\boldsymbol{v}) \leq \deg \Phi_{\boldsymbol{w}}(\boldsymbol{Q})$. But then if $t = \mathrm{LP}(\Phi_{\boldsymbol{w}}(\boldsymbol{Q}))$ we have that for any i:

$$\deg\left(v_i x^{i(k-1)}\right) \leq \deg \Phi_{\boldsymbol{w}}(\boldsymbol{Q}) = \deg\left(Q_t x^{t(k-1)}\right) < \chi$$

Hence, $\boldsymbol{v}$ satisfies (4). □

This results immediately in the decoding procedure outlined as Algorithm 2.

Algorithm 2 MV interpolation step by row reduction

```
Input: An instance of Problem 2
Output: A vector $\boldsymbol{Q} \in \mathcal{R}^{\ell+1}$ solving Problem 2.
1 Set up $M$ as in 6.
2 Compute a $\boldsymbol{w}$-shifted weak Popov form $V$ of $M$.
3 return the row $\boldsymbol{v}$ of $V$ which has minimal $\boldsymbol{w}$-shifted degree $\deg \Phi_{\boldsymbol{w}}(\boldsymbol{v})$.
```

Theorem 4 *Algorithm 2 has complexity $O(\ell n^2)$ over $\mathbb{F}_{q^s}$.*

Proof Computing G can be done straightforwardly in $O(n^2)$ operations over $\mathbb{F}_{q^s}$. Each R_t can be computed in the same speed using a decomposition into smaller interpolations and two annihilator polynomials, see e.g. [39]. For Line 2, we use Algorithm 7 whose complexity is $O(\ell n^2)$, proved as Theorem 10. □

In [48], Xie, Lin, Yan and Suter present an algorithm for solving the Interpolation Step using a skew-variant of the Kötter–Nielsen–Høholdt algorithm [35] with complexity $O(\ell^2 sn)$ over $\mathbb{F}_{q^s}$. Since $n < s$, our algorithm is at least as fast as theirs. Note that these costs probably dominate the complexity of MV decoding: the other step, Root-finding, likely[5] has complexity $O(\ell^2 kn)$.

4 Row reduction of $\mathcal{R}$-matrices

4.1 The Mulders–Storjohann algorithm

In this section, we introduce our algorithmic work horse: obtaining row reduced bases of left $\mathcal{R}$-modules $\mathcal{V} \subseteq \mathcal{R}^m$. The core is an $\mathcal{R}$-variant of the Mulders–Storjohann algorithm [31] that was originally described for $\mathbb{F}[x]$ matrices. The algorithm and its proof of correctness carries over almost unchanged, while a fine-grained complexity analysis is considerably more involved; we return to this in Sect. 4.3.

[5] In [28], the claimed complexity of their root-finding is $O(\ell^{O(1)}k)$. However, we have to point out that the complexity analysis of that algorithm has severe issues which are outside the scope of this paper to amend. There are two problems: (1) It is not proven that the recursive calls will not produce many spurious "pseudo-roots" which are sifted away only at the leaf of the recursions; and (2) the cost analysis ignores the cost of computing the shifts $Q(X, Y^q + \gamma Y)$. Issue 1 is necessary to resolve for assuring polynomial complexity. For the original $\mathbb{F}[x]$-algorithm this is proved as [42, Proposition 6.4], and an analogous proof might carry over. Issue 2 is critical since these shifts dominate the complexity: assuming the algorithm makes a total of $O(\ell k)$ recursive calls to itself, then $O(\ell k)$ shifts need to be computed, each of which costs $O(\ell \deg_x Q) \subset O(\ell n)$. If Issue 1 is resolved the algorithm should then have complexity $O(\ell^2 kn)$.

Algorithm 3 Mulders–Storjohann for $\mathcal{R}$ matrices

```
Input: A matrix $V$ over $\mathcal{R}$, whose rows span a module $\mathcal{V}$.
Output: A basis of $\mathcal{V}$ in weak Popov form.
1 Until no longer possible, apply a simple LP-transformation on two rows in $V$.
2 return $V$.
```

Definition 5 Applying a *simple transformation i on j at position h* on a matrix V with $\deg v_{i,h} \leq \deg v_{j,h}$ means to replace $\boldsymbol{v}_j$ by $\boldsymbol{v}_j - \alpha x^{\beta} \boldsymbol{v}_i$, where $\beta = \deg v_{j,h} - \deg v_{i,h}$ and $\alpha = \mathrm{LC}(v_{j,h})/\theta^{\beta}(\mathrm{LC}(v_{i,h}))$.

By a *simple LP-transformation i on j*, where $\mathrm{LP}(\boldsymbol{v}_i) = \mathrm{LP}(\boldsymbol{v}_j)$, we will mean a simple transformation i on j at position $\mathrm{LP}(\boldsymbol{v}_i)$.

Remark 1 Note that a simple transformation i on j at position h cancels the leading term of the polynomial $v_{j,h}$. Elementary row operations keep the row space and rank of the matrix unchanged, and in particular so does any sequence of simple transformations.

We use the following *value function* for $\mathcal{R}$ vectors as a "size" of $\mathcal{R}^m$ vectors:

$$\psi : \mathcal{R}^m \to \mathbb{Z}_{\geq 0}$$
$$\boldsymbol{v} \mapsto \begin{cases} 0 & \text{if } \boldsymbol{v} = \boldsymbol{0} \\ m \deg \boldsymbol{v} + \mathrm{LP}(\boldsymbol{v}) + 1 & \text{otherwise} \end{cases}$$

Lemma 6 *For some $V \in \mathcal{R}^{\cdot \times m}$, consider a simple LP-transformation i on j, where $\boldsymbol{v}_j$ is replaced by $\boldsymbol{v}'_j$. Then $\psi(\boldsymbol{v}'_j) < \psi(\boldsymbol{v}_j)$.*

Proof The proof works exactly as in the $\mathbb{F}[x]$ case, cf. [32, Lemma 8]. □

Theorem 5 *Algorithm 3 is correct.*

Proof By Lemma 6, the ψ-value of one row of V decreases for each simple LP-transformation. The sum of the values of the rows must at all times be non-negative so the algorithm must terminate. When the algorithm terminates there are no $i \neq j$ such that $\mathrm{LP}(\boldsymbol{v}_i) = \mathrm{LP}(\boldsymbol{v}_j)$. That is to say, V is in weak Popov form. □

The above proof easily leads to the rough complexity estimate of Algorithm 3 of $O(m^2 \deg V \operatorname{maxdeg} V)$, where m is the number of columns in V.

Note that in Algorithm 3 each iteration might present several possibilities for the simple LP-transformation; the above theorem shows that any choice of LP-transformations leads to the correct result.

To transform V into $\boldsymbol{w}$-shifted weak Popov form, for some shift $\boldsymbol{w} \in \mathbb{Z}^m_{\geq 0}$, we let $V' = \Phi_{\boldsymbol{w}}(V)$ and apply Algorithm 3 on V' to obtain W' in weak Popov form. Since Algorithm 3 only performs row operations, it is clear that $\Phi_{\boldsymbol{w}}$ can be inverted on W' to obtain $W = \Phi_{\boldsymbol{w}}^{-1}(W')$. Then W is in $\boldsymbol{w}$-shifted weak Popov form by definition.

4.2 The determinant degree and orthogonality defect

The purpose of this section is to introduce the orthogonality defect as a tool for measuring "how far" a square, full-rank matrix over $\mathcal{R}$ is from being in weak Popov form. It relies on the nice properties of the degree of the Dieudonné determinant for matrices over $\mathcal{R}$. The orthogonality defect for $\mathbb{F}[x]$ matrices was introduced by Lenstra [22] and used in [32] to similar effect as we do here.

Dieudonné introduced a function for matrices over skew fields which shares some of the essential properties of the usual commutative determinant, in particular that it is multiplicative, see [12] or [13, Sect. 20]. This Dieudonné determinant can be applied to matrices over $\mathcal{R}$ by considering $\mathcal{R}$ inside its left field of fractions. The definition of this determinant is quite technical, and we will not actually need to invoke it. Rather, we will use an observation by Taelman [46] that the Dieudonné determinant implies a simple-behaving *determinant degree* function for matrices with very nice properties:

Proposition 1 *There is a unique function* $\deg\det : \mathcal{R}^{m\times m} \to \mathbb{Z}_{\geq 0} \cup \{-\infty\}$ *s.t.:*

- $\deg\det(AA') = \deg\det(A) + \deg\det(A')$ *for all* $A, A' \in \mathcal{R}^{m\times m}$.
- $\deg\det U = 0$ *for all* $U \in GL_m(\mathcal{R})$.
- *If* A *is diagonal with diagonal elements* $d_0, \dots, d_{m-1}$, *then* $\deg\det A = \sum_i \deg d_i$

Corollary 1 *For any* $A, A' \in \mathcal{R}^{m\times m}$ *then:*

- *If* A' *is obtained from* A *by elementary row operations, then* $\deg\det A' = \deg\det A$.
- *If* B *equals* $A \in \mathcal{R}^{m\times m}$ *with one row or column scaled by some* $f \in \mathcal{R}^*$, *then* $\deg\det B = \deg f + \deg\det A$.
- *If* A *is triangular with diagonal elements* $d_0, \dots, d_{m-1}$, *then* $\deg\det A = \sum_i \deg d_i$.
- $\deg\det(\Phi_{\boldsymbol{w}}(A)) = \deg\det(A) + \sum_i w_i$ *for any shift* $\boldsymbol{w}$.

Example 1 Consider input matrix $\Phi_{\boldsymbol{w}}(M)$ to Algorithm 1 for the case[6] $\ell = 2$, $\boldsymbol{w} = (\gamma_0, \gamma_1, \gamma_2) = (100, 42, 69)$, $\deg s_1 = 99$, $\deg s_2 = 95$ and $\deg g_1 = \deg g_2 = 100$. Then

$$\Phi_{\boldsymbol{w}}(M) = \begin{pmatrix} x^{\gamma_0} & s_1 x^{\gamma_1} & s_2 x^{\gamma_2} \\ & g_1 x^{\gamma_1} & \\ & & g_2 x^{\gamma_2} \end{pmatrix} = \begin{pmatrix} 1 & s_1 & s_2 \\ & g_1 & \\ & & g_2 \end{pmatrix} \begin{pmatrix} x^{\gamma_0} & & \\ & x^{\gamma_1} & \\ & & x^{\gamma_2} \end{pmatrix}.$$

And so by Proposition 1, $\deg\det \Phi_{\boldsymbol{w}}(M) = \deg g_1 + \deg g_2 + \sum_i \gamma_i = 411$.

This description of $\deg\det(\cdot)$ is not operational in the sense that it is not clear how to compute $\deg\det V$ for general $V \in \mathcal{R}^{m\times m}$. The following definition and Proposition 2 implies that Algorithm 3 can be used to compute $\deg\det V$; conversely, we show in Sect. 4.3 how to bound the complexity of Algorithm 3 based on $\deg\det V$.

Definition 6 The orthogonality defect of $V \in \mathcal{R}^{m\times m}$ is $\Delta(V) := \deg V - \deg\det V$.

The following observations are easy for $\mathbb{F}[x]$ matrices, but require more work over $\mathcal{R}$:

Proposition 2 *Let* $V \in \mathcal{R}^{m\times m}$ *of full rank and in weak Popov form. Then* $\Delta(V) = 0$.

Proof Due to Corollary 1, we can assume the columns and rows of V are ordered such that $\mathrm{LP}(\boldsymbol{v}_i) = i$ and $\deg v_{i,i} \leq \deg v_{j,j}$ for $i < j$. We will call this property "ordered weak Popov form" in this proof. Note that it implies $\psi(\boldsymbol{v}_i) < \psi(\boldsymbol{v}_j)$ for $i < j$. We will inductively obtain a series of matrices $V^{(0)} = V, V^{(1)}, V^{(2)}, \dots, V^{(m)}$ each in ordered weak Popov form, and such that the first i columns of $V^{(i)}$ are zero below the diagonal. Then $V^{(m)}$ is upper triangular and we can obtain two expressions for its deg det.

6 This is a realistic shift register problem arising in decoding of an Interleaved Gabidulin code with $n = s = 100$, $k_1 = 58$, $k_2 = 31$.

So assume that $V^{(i)}$ is in ordered weak Popov form and its first i columns are zero below the diagonal. Recall that the (left) *union* of two skew polynomials $f, g \in \mathcal{R}$ is the unique lowest-degree $p \in \mathcal{R}$ such that $p = af = bg$ for some $a, b \in \mathcal{R}$; it is a consequence of the Euclidean algorithm that the union always exists, see e.g. [38]. For each $j > i$ consider now the coefficients in the union of $v_{i,i}^{(i)}$ and $v_{j,i}^{(i)}$, i.e. $a_j^{(i)}, b_j^{(i)} \in \mathcal{R}$ such that $a_j^{(i)} v_{i,i}^{(i)} = b_j^{(i)} v_{j,i}^{(i)}$. Let

$$V^{(i+1)} = \left(\begin{array}{c|cccc} I_{i-1} & & & & \\ \hline & 1 & & & \\ & -a_{i+1}^{(i)} & b_{i+1}^{(i)} & & \\ & \vdots & & \ddots & \\ & -a_{m-1}^{(i)} & & & b_{m-1}^{(i)} \end{array}\right) V^{(i)} ,$$

where I_{i-1} is the $(i-1) \times (i-1)$ identity matrix. The $i+1$ first columns of $V^{(i+1)}$ are then zero below the diagonal. Also $\mathrm{LP}(a_j^{(i)} \boldsymbol{v}_i^{(i)}) < \mathrm{LP}(b_j^{(i)} \boldsymbol{v}_j^{(i)}) = \mathrm{LP}(\boldsymbol{v}_j^{(i)})$ and $\deg(a_j^{(i)} \boldsymbol{v}_i^{(i)}) \leq \deg(b_j^{(i)} \boldsymbol{v}_j^{(i)})$ for $j > i$, which means $\psi(a_j^{(i)} \boldsymbol{v}_i^{(i)}) < \psi(b_j^{(i)} \boldsymbol{v}_j^{(i)})$ and therefore $\psi(\boldsymbol{v}_j^{(i+1)}) = \psi(b_j^{(i)} \boldsymbol{v}_j^{(i)})$. This implies that $V^{(i+1)}$ is in ordered weak Popov form and that $\deg v_{j,j}^{(i+1)} = \deg b_j^{(i)} + \deg v_{j,j}^{(i)}$ for $j > i$, which inductively expands to

$$\deg v_{j,j}^{(i+1)} = \deg v_{j,j} + \sum_{h=0}^{i} \deg b_j^{(h)} .$$

Inductively, we therefore arrive at an upper triangular matrix $V^{(m)}$ in ordered weak Popov form, and whose diagonal elements satisfy $\deg v_{j,j}^{(m)} = \deg v_{j,j} + \sum_{i=0}^{j-1} \deg b_j^{(i)}$. Thus $\deg\det V^{(m)}$ is the sum of all these degrees by Corollary 1. On the other hand $V^{(m)}$ is obtained by multiplying triangular matrices on $V^{(0)} = V$, so by Proposition 1 we get another expression for $\deg\det V^{(m)}$ as:

$$\deg\det V^{(m)} = \deg\det V + \sum_{i=0}^{m-1} \sum_{j=i+1}^{m-1} \deg b_j^{(i)}$$

Combining the expressions, we get $\deg\det V = \sum_{i=0}^{m-1} \deg v_{i,i} = \deg V$. □

Corollary 2 *Let $V \in \mathcal{R}^{m \times m}$ and full-rank, then $0 \leq \deg\det V \leq \deg V$.*

Proof Applying Algorithm 3 on V would use row operations to obtain a matrix $V' \in \mathcal{R}^{m \times m}$ in weak Popov form. Then $\deg\det V = \deg\det V'$ by Proposition 1. By Proposition 2 then $\deg\det V' = \deg V' \geq 0$, and by the nature of Algorithm 3, then $\deg V' \leq \deg V$. □

4.3 Complexity of Mulders–Storjohann

We can now bound the complexity of Algorithm 3 using arguments completely analogous to the $\mathbb{F}[x]$ case in [32]. These are in turn, the original arguments of [31] but finer grained by using the orthogonality defect. We bring the full proof here since the main steps are referred to in Sect. 5.1.

Theorem 6 *Algorithm 3 with a full-rank input matrix $V \in \mathcal{R}^{m \times m}$ performs at most $m\big(\Delta(V) + m\big)$ simple LP-transformations, and it has complexity $O(m^2 \Delta(V) \operatorname{maxdeg}(V))$ over $\mathbb{F}$.*

Proof By Lemma 6, every simple LP-transformation reduces the ψ-value of one row with at least 1. So the number of possible simple LP-transformations is upper bounded by the difference of values of the input matrix V and the output matrix U, the matrices values being the sum of their rows'. More precisely, the number of iterations is upper bounded by:

$$\begin{aligned} &\textstyle\sum_{i=0}^{m-1}\left[m\deg \boldsymbol{v}_i + \mathrm{LP}(\boldsymbol{v}_i) - \left(m\deg \boldsymbol{u}_i + \mathrm{LP}(\boldsymbol{u}_i)\right)\right] \\ &\le m^2 + m\textstyle\sum_{i=0}^{m-1}\left[\deg \boldsymbol{v}_i - \deg \boldsymbol{u}_i\right] \\ &= m[\deg V - \deg U + m] = m(\Delta(V)+m), \end{aligned}$$

where the last equality follows from $\deg U = \deg\det U$ due to Proposition 2 and $\deg\det U = \deg\det V$.

One simple transformation consists of calculating $\boldsymbol{v}_j - \alpha x^\beta \boldsymbol{v}_i$, so for every coefficient in $\boldsymbol{v}_i$, we must apply θ^β, multiply by α and then add it to a coefficient in $\boldsymbol{v}_j$, each being in $O(1)$. Since $\deg \boldsymbol{v}_j \le \mathrm{maxdeg}(V)$ this costs $O(m\,\mathrm{maxdeg}(V))$ operations in $\mathbb{F}$. □

Since $\Delta(V) \le \deg V$, the above complexity bound is always at least as good as the straightforward bound we mentioned at the end of Sect. 4.1.

Example 2 [Mulders–Storjohann algorithm on an MgLSSR] Consider an instance of Problem 1. The complexity of Algorithm 1 is determined by a row reduction of

$$\Phi_{\boldsymbol{w}}(M) = \begin{pmatrix} x^{\gamma_0} & s_1x^{\gamma_1} & s_2x^{\gamma_2} & \dots & s_\ell x^{\gamma_\ell} \\ & g_1x^{\gamma_1} & & & \\ & & g_2x^{\gamma_2} & & \\ & & & \ddots & \\ & & & & g_\ell x^{\gamma_\ell} \end{pmatrix}. \tag{7}$$

Let $\mu := \max_i\{\gamma_i + \deg g_i\}$. We can assume that $\gamma_0 < \max_{i\ge1}\{\gamma_i + \deg s_i\} \le \mu$ since otherwise M is already in $\boldsymbol{w}$-shifted weak Popov form. To apply Theorem 6, we calculate the orthogonality defect of $\Phi_{\boldsymbol{w}}(M)$. Since it is upper triangular, the degree of its determinant is

$$\deg\det \Phi_{\boldsymbol{w}}(M) = \sum_{i=1}^{\ell}\deg g_i + \sum_{i=0}^{\ell}\gamma_i\ .$$

The degrees of the rows of $\Phi(M)$ satisfy

$$\begin{aligned} \deg \Phi_{\boldsymbol{w}}(\boldsymbol{m}_0) &= \max_i\{\gamma_i + \deg s_i\} \le \mu\ , \\ \deg \Phi_{\boldsymbol{w}}(\boldsymbol{m}_i) &= \gamma_i + \deg g_i \qquad\qquad \text{for } i \ge 1. \end{aligned}$$

Thus, $\Delta(\Phi_{\boldsymbol{w}}(M)) \le \mu - \gamma_0$. With $\mathrm{maxdeg}(\Phi_{\boldsymbol{w}}(M)) \le \mu$, Theorem 6 implies a complexity of $O(\ell^2\mu^2)$, assuming $\ell \in O(\mu)$. Note that the straightforward bound on Algorithm 3 yields $O(\ell^3\mu^2)$. □

Example 3 (Mulders–Storjohann for the Interpolation Step in decoding MV codes) Line 2 of Algorithm 2 is a row reduction of $\Phi_{\boldsymbol{w}}(M)$, as defined in (6) on page 8, whose degrees of the nonzero entries are component-wise upper bounded by:

$$\begin{pmatrix} n & & & & \\ n & (k-1) & & & \\ n & & 2(k-1) & & \\ \vdots & & & \ddots & \\ n & & & & \ell(k-1) \end{pmatrix}$$

Thus $\Delta(\Phi_{\boldsymbol{w}}(M)) \leq \deg \Phi_{\boldsymbol{w}}(M) - n - \binom{\ell+1}{2}(k-1) \leq \ell n$. Using Theorem 6, the complexity in operations over $\mathbb{F}_{q^s}$ becomes $O(\ell^3 n^2)$.

5 Faster row reduction on matrices having special forms

In this section, we will investigate improved row reduction algorithms for matrices of special forms. The main goals are to improve the running time of row reducing the matrices appearing in the decoding settings of Sects. 3.1 and 3.2, but the results here apply more broadly.

5.1 Shift register problems: the demand-driven algorithm

Our first focus is to improve the MgLSSR case of Algorithm 1 on page 7, where we are to row reduce $\Phi_{\boldsymbol{w}}(M)$, given by (7): Algorithm 4 is a refinement of Algorithm 3 which is asymptotically faster when all g_i are of the form $x^{d_i} + a_i$ for $a_i \in \mathbb{F}$. Though the refinement is completely analogous to that of [32] for the $\mathbb{F}[x]$ case, no complete proof has appeared in unabridged, peer-reviewed form before, so we give full proofs of the $\mathcal{R}$ case here. We begin with a technical lemma:

Lemma 7 *Consider an instance of Problem 1 and Algorithm 3 with input $\Phi_{\boldsymbol{w}}(M)$ of (7). Let $\tilde{g}_j = g_j x^{\gamma_j}$. Consider a variant of Algorithm 3 where, after a simple LP-transformation i on j, which replaces $\boldsymbol{v}_j$ with $\boldsymbol{v}'_j$, we instead replace it with $\boldsymbol{v}''_j = (v'_{j,0}, v'_{j,1} \bmod \tilde{g}_1, \ldots, v'_{j,\ell} \bmod \tilde{g}_\ell)$. This does not change the correctness of the algorithm or the upper bound on the number of simple LP-transformations performed.*

Proof Correctness follows if we can show that each of the ℓ modulo reductions could have been achieved by a series of row operations on the current matrix V after the simple LP-transformation producing $\boldsymbol{v}'_j$. For each $h \geq 1$, let $\boldsymbol{g}_h = (0, \ldots, 0, \tilde{g}_h, 0, \ldots, 0)$, with position h non-zero.

During the algorithm, we will let J_h be a subset of the current rows in V having two properties: that $\boldsymbol{g}_h$ can be constructed as an $\mathcal{R}$-linear combination of the rows in J_h; and that each $\boldsymbol{v} \in J_h$ has $\psi(\boldsymbol{v}) \leq \psi(\boldsymbol{g}_h)$. Initially, $J_h = \{\boldsymbol{g}_h\}$.

After simple LP-transformations on rows not in J_h, the h'th modulo reduction is therefore allowed, since $\boldsymbol{g}_h$ can be constructed by the rows in J_h. On the other hand, consider a simple LP-transformation i on j where $\boldsymbol{v}_j \in J_h$, resulting in the row $\boldsymbol{v}'_j$. Then the h'th modulo reduction has no effect since $\psi(\boldsymbol{v}'_j) < \psi(\boldsymbol{v}_j) \leq \psi(\boldsymbol{g}_h)$. Afterwards, J_h is updated as $J_h = J_h \setminus \{\boldsymbol{v}_j\} \cup \{\boldsymbol{v}'_j, \boldsymbol{v}_i\}$. We see that J_h then still satisfies the two properties, since $\psi(\boldsymbol{v}_i) \leq \psi(\boldsymbol{v}_j) \leq \psi(\boldsymbol{g}_h)$.

Since $\psi(\boldsymbol{v}''_j) \leq \psi(\boldsymbol{v}'_j)$ the proof of Theorem 6 shows that the number of simple LP-transformations performed is still bounded by $(\ell+1)(\Delta(V) + \ell + 1)$. □

Theorem 7 *Algorithm 4 is correct.*

Proof We first prove that an intermediary algorithm, Algorithm 5, is correct using the correctness of Algorithm 3, and then prove the correctness of Algorithm 4 using Algorithm 5 and Lemma 7. Starting from Algorithm 3 with input $\Phi_{\boldsymbol{w}}(M)$, then Algorithm 5 is obtained by two simple modifications: Firstly, note that initially, when $V := \Phi_{\boldsymbol{w}}(M)$, then $\mathrm{LP}(\boldsymbol{v}_h) = h$ for $h \geq 1$, and therefore the only possible simple LP-transformation must involve $\boldsymbol{v}_0$. We can maintain this property as a loop invariant throughout the algorithm by swapping $\boldsymbol{v}_0$ and $\boldsymbol{v}_{\mathrm{LP}(\boldsymbol{v}_0)}$ when applying a simple LP-transformation $\mathrm{LP}(\boldsymbol{v}_0)$ on 0.

Algorithm 4 Demand-Driven algorithm for MgLSSR

```
Input: Instance of Problem 1. $\tilde{s}_j \leftarrow s_{1,j}x^{\gamma_j}$, $\tilde{g}_j \leftarrow g_j x^{\gamma_j}$ for $j = 1, \ldots, \ell$.
Output: The zeroth column of a basis of $\mathcal{M}$ of (1) in $\boldsymbol{w}$-shifted weak Popov form.
1  $(\eta, h) \leftarrow$ (deg, LP) of $(x^{\gamma_0}, \tilde{s}_1, \ldots, \tilde{s}_\sigma)$.
2  if $h = 0$ then return $(1, 0, \ldots, 0)$.
3  $(\lambda_0, \ldots, \lambda_\ell) \leftarrow (x^{\gamma_0}, 0, \ldots, 0)$.
4  $\alpha_j x^{\eta_j} \leftarrow$ the leading monomial of $\tilde{g}_j$ for $j = 1, \ldots, \ell$.
5  while $\deg \lambda_0 \le \eta$ do
6      $\alpha \leftarrow$ coefficient to $x^\eta$ in $(\lambda_0 \tilde{s}_h \mod \tilde{g}_h)$.
7      if $\alpha \neq 0$ then
8          if $\eta < \eta_h$ then swap $(\lambda_0, \alpha, \eta)$ and $(\lambda_h, \alpha_h, \eta_h)$.
9          $\lambda_0 \leftarrow \lambda_0 - \alpha/\theta^{\eta-\eta_h}(\alpha_h) x^{\eta-\eta_h} \lambda_h$.
10     $(\eta, h) \leftarrow (\eta, h-1)$ if $h > 1$ else $(\eta - 1, \ell)$.
11 return $\left(\lambda_0 x^{-\eta_0}, \ldots, \lambda_\ell x^{-\eta_0}\right)$.
```

Intermediate Algorithm 5 for the correctness proof of Algorithm 4

```
Input: Instance of Problem 1. $V \leftarrow \Phi_{\boldsymbol{w}}(M)$ with $M$ as in (7).
Output: A basis $V'$ of $\mathcal{M}$ of (1) in $\boldsymbol{w}$-shifted weak Popov form.
1  $(\eta, h) \leftarrow$ (deg, LP) of $\boldsymbol{v}_0$.
2  if $h = 0$ then return $\Phi_{\boldsymbol{w}}^{-1}(V)$.
3  while $\deg v_{0,0} \le \eta$ do
4      $\alpha \leftarrow$ coefficient to $x^\eta$ in $v_{0,h}$.
5      if $\alpha \neq 0$ then
6          $\eta_h \leftarrow \deg \boldsymbol{v}_h$.
7          $\alpha_h \leftarrow$ coefficient to $x^{\eta_h}$ in $v_{h,h}$.
8          if $\eta < \eta_h$ then swap $(\boldsymbol{v}_0, \alpha, \eta)$ and $(\boldsymbol{v}_h, \alpha_h, \eta_h)$.
9          $\boldsymbol{v}_0 \leftarrow \boldsymbol{v}_0 - \alpha/\theta^{\eta-\eta_h}(\alpha_h) x^{\eta-\eta_h} \boldsymbol{v}_h$.
10     $(\eta, h) \leftarrow (\eta, h-1)$ if $h > 1$ else $(\eta - 1, \ell)$.
11 return $\Phi_{\boldsymbol{w}}^{-1}(V)$.
```

The second modification is to maintain (η, h) as an upper bound on the (deg, LP) of $\boldsymbol{v}_0$ throughout the algorithm: we initially simply compute these values. Whenever we have applied a simple LP-transformation on $\boldsymbol{v}_0$ resulting in $\boldsymbol{v}_0'$, we know by Lemma 6 that $\psi(\boldsymbol{v}_0') < \psi(\boldsymbol{v}_0)$. Therefore, either $\deg \boldsymbol{v}_0' < \eta$ or $\deg \boldsymbol{v}_0' = \eta \wedge \mathrm{LP}(\boldsymbol{v}_0') < h$. This is reflected in a corresponding decrement of (η, h).

As a loop invariant we therefore have $\psi(\boldsymbol{v}_0) \le \eta(\ell + 1) + h$. After an iteration, if this inequality is sharp, it simply implies that the α computed in the following iteration will be 0, and (η, h) will be correspondingly decremented once more. Note that we never set $h = 0$: when $\mathrm{LP}(\boldsymbol{v}_0) = 0$ then V must be in weak Popov form (since we already maintain $\mathrm{LP}(\boldsymbol{v}_h) = h$ for $h > 0$). At this point, the while-loop will be exited since $\deg v_0 > \eta$.

Algorithm 5 is then simply the implementation of these modifications, and writing out in full what the simple LP-transformation does to $\boldsymbol{v}_0$. This proves that Algorithm 5 is operationally equivalent to Algorithm 3 with input $\Phi_{\boldsymbol{w}}(M)$.

For obtaining Algorithm 4 from Algorithm 5, the idea is to store only the necessary part of V and compute the rest on demand. Firstly, by Lemma 7 correctness would be maintained if the simple LP-transformation on Line 9 of Algorithm 5 was followed by the ℓ modulo reductions. In that case, we would have $v_{0,h} = (v_{0,0}\tilde{s}_h \mod \tilde{g}_h)$, so storing only $v_{0,0}$ suffice for reconstructing $\boldsymbol{v}_0$. Consequently we store the first column of V in Algorithm 4 as $(\lambda_0, \ldots, \lambda_\ell)$. Line 6 of Algorithm 4 is now the computation of the needed coefficient of $v_{0,h}$ at the latest possible time.

As $\deg \boldsymbol{v}_h$ is used in Line 6 of Algorithm 5, we need to store and maintain this between iterations; this is the variables $\eta_1, \ldots, \eta_\ell$. To save some redundant computation of coefficients, the x^{η_h}-coefficient of $v_{h,h}$ is also stored as α_h.

This proves that Algorithm 4 is operationally equivalent to Algorithm 5, which finishes the proof of correctness. □

Proposition 3 *Algorithm 4 has computational complexity $O(\ell\mu^2 + \sum_{h=1}^{\ell}\sum_{\eta=0}^{\mu-1} T_{h,\eta})$, where $\mu = \max_i\{\gamma_i + \deg g_i\}$ and $T_{h,\eta}$ bounds the complexity of running Line 6 for those values of h and η.*

Proof Clearly, all steps of the algorithm are essentially free except Lines 6 and 9. Observe that every iteration of the while-loop decreases an *upper bound* on the value of row 0, whether we enter the if-branch in Line 7 or not. So by the arguments of the proof of Theorem 6, the loop will iterate at most $O(\ell\mu)$ times in which each possible value of $(h, \eta) \in \{1, \ldots, \ell\} \times \{0, \ldots, \mu - 1\}$ will be taken at most once. Each execution of Line 9 costs $O(\mu)$ since the λ_j all have degree at most μ. □

It is possible to use Proposition 3 to show that Algorithm 4 is efficient if e.g. all the g_i have few non-zero monomials.[7] We will restrict ourselves to a simpler case which nonetheless has high relevance for coding theory:

Theorem 8 *Algorithm 4 can be realised with complexity $O(\ell\mu^2)$ if $g_i = x^{d_i} + a_i$ for $a_i \in \mathbb{F}_q$ for all i, where $\mu = \max_i\{\gamma_i + \deg g_i\}$.*

Proof We will bound $\sum_{\eta=0}^{\mu-1} T_{h,\eta}$ of Proposition 3. Note first that for any η, the coefficient α to x^η in $(\lambda\tilde{s}_h \bmod \tilde{g}_h)$ equals the coefficient to $x^{\eta-\gamma_h}$ of $(\lambda s_h \bmod g_h)$, so considering $\gamma_h = 0$ suffice. Now if $\eta \geq d_h$ then $\alpha = 0$ and can be returned immediately. If $\eta < d_h$, then due to the assumed shape of g_i, α is a linear combination of the coefficients to $x^\eta, x^{\eta+d_h}, \ldots, x^{\eta+td_h}$ in $\lambda_0 s_h$, where $t = \left\lfloor \frac{\mu-\eta}{d_h} \right\rfloor$. Each such coefficient can be computed by convolution of λ_0 and s_h in $O(\mu)$, so it costs $O(\frac{\mu^2}{d})$ to compute α. Summing over all choices of η, we have $\sum_{\eta=0}^{\mu-1} T_{h,\eta} \in O(\mu^2)$ and the theorem follows from Proposition 3. □

5.2 Weak Popov walking

The goal of this section is to arrive at a faster row reduction algorithm for the matrices used for decoding MV codes in Sect. 3.2. However, the algorithm we describe could be of much broader interest: it is essentially an improved way of computing a $\boldsymbol{w}$-weak Popov form of a matrix which is already in $\boldsymbol{w}'$-weak Popov form, for a shift $\boldsymbol{w}'$ which is not too far from $\boldsymbol{w}$. Inspired by "Gröbner walks", we have dubbed this strategy "weak Popov walking". Each "step" of the walk can be seen as just Algorithm 3 but where we carefully choose which LP-transformations to apply each iteration, in case there is choice.

This strategy would work completely equivalently for the $\mathbb{F}[x]$ case. However, to the best of our knowledge, that has not been done before.

In this section we will extensively discuss vectors under different shifts. To ease the notation we therefore introduce shifted versions of the following operators: $\mathrm{LP}_{\boldsymbol{w}}(\boldsymbol{v}) := \mathrm{LP}(\Phi_{\boldsymbol{w}}(\boldsymbol{v}))$ as well as $\deg_{\boldsymbol{w}}(\boldsymbol{v}) := \deg \Phi_{\boldsymbol{w}}(\boldsymbol{v})$.

[7] In the conference version of this paper [24], we erroneously claimed a too strong statement concerning this. However, one *can* relate the complexity of Algorithm 4 to the number of non-zero monomials of g_i, as long as all but the leading monomial have low degree; however the precise statement becomes cumbersome and is not very relevant for this paper.

We begin by Algorithm 6 that efficiently "walks" from a weak Popov form according to the shift $\boldsymbol{w}$ into one with the shift $\boldsymbol{w} + (1, 0, \dots, 0)$. The approach can readily be generalised to support increment on any index, but we do not need it for the decoding problem so we omit the generalisation to simplify notation.

Algorithm 6 Weak Popov walking

```
Input: Shift $\boldsymbol{w} \in \mathbb{Z}_{\geq 0}^{m}$ and matrix $V \in \mathcal{R}^{m \times m}$ in $\boldsymbol{w}$-shifted weak Popov form.
Output: Matrix in $\hat{\boldsymbol{w}}$-shifted weak Popov form spanning the same $\mathcal{R}$-row space as $V$, where $\hat{\boldsymbol{w}} = \boldsymbol{w} + (1, 0, \dots, 0)$.
1  $h_i \leftarrow \mathrm{LP}_{\boldsymbol{w}}(\boldsymbol{v}_i)$, for $i = 0, \dots, m-1$.
2  $I \leftarrow$ indexes $i$ such that $\mathrm{LP}_{\hat{\boldsymbol{w}}}(\boldsymbol{v}_i) = 0$.
3  $[i_1, \dots, i_s] \leftarrow I$ sorted such that $h_{i_1} < h_{i_2} < \dots < h_{i_s}$.
4  $t \leftarrow i_1$.
5  for $i = i_2, \dots, i_s$ do
6      if $\deg v_{t,0} \leq \deg v_{i,0}$ then
7          Apply a simple transformation $t$ on $i$ at position 0 in $V$.
8      else
9          Apply a simple transformation $i$ on $t$ at position 0 in $V$.
10         $t \leftarrow i$.
11 return $V$.
```

Theorem 9 *Algorithm 6 is correct.*

Proof Denote in this proof V as the input and $\hat{V}$ as the output of the algorithm. The algorithm performs a single sweep of simple transformations, modifying only rows indexed by I: in particular, if $\boldsymbol{v}_i$, $\hat{\boldsymbol{v}}_i$ are the rows of V respectively $\hat{V}$, then either $\hat{\boldsymbol{v}}_i = \boldsymbol{v}_i$, or $\hat{\boldsymbol{v}}_i$ is the result of a simple transformation on $\boldsymbol{v}_i$ by another row $\boldsymbol{v}_j$ of V and $i, j \in I$. All the h_i are different since V is in $\boldsymbol{w}$-shifted weak Popov form. We will show that the $\hat{\boldsymbol{w}}$-shifted leading positions of $\hat{V}$ is a permutation of the h_i, implying that $\hat{V}$ is in $\hat{\boldsymbol{w}}$-shifted weak Popov form.

Note first that for any vector $\boldsymbol{v} \in \mathcal{R}^m$ with $\mathrm{LP}_{\boldsymbol{w}}(\boldsymbol{v}) \neq \mathrm{LP}_{\hat{\boldsymbol{w}}}(\boldsymbol{v})$, then $\mathrm{LP}_{\hat{\boldsymbol{w}}}(\boldsymbol{v}) = 0$, since only the degree of the 0'th position of $\Phi_{\hat{\boldsymbol{w}}}(\boldsymbol{v})$ is different from the corresponding position of $\Phi_{\boldsymbol{w}}(\boldsymbol{v})$. For each $i \in \{0, \dots, m-1\} \setminus I$ we have $\hat{\boldsymbol{v}}_i = \boldsymbol{v}_i$ and so $\mathrm{LP}_{\hat{\boldsymbol{w}}}(\hat{\boldsymbol{v}}_i) = h_i$. And of course for each $i \in I$ we have $\mathrm{LP}_{\hat{\boldsymbol{w}}}(\boldsymbol{v}_i) = 0$. This implies for each $j \in I$ that:

$$\deg v_{j,0} + w_0 = \deg v_{j,h_j} + w_{h_j} \ . \tag{8}$$

Consider first an index $i \in I$ for which Line 9 was run, and let t be as at that point. This means $\hat{\boldsymbol{v}}_i = \boldsymbol{v}_i + \alpha x^{\delta} \boldsymbol{v}_t$ for some $\alpha \in \mathbb{F}$ and $\delta = \deg v_{i,0} - \deg v_{t,0}$. Note that the if-condition ensures $\delta \geq 0$ and the simple transformation makes sense. We will establish that $\mathrm{LP}_{\hat{\boldsymbol{w}}}(\hat{\boldsymbol{v}}_i) = h_i$. Since we are performing an LP-transformation, we know that $\deg_{\hat{\boldsymbol{w}}} \hat{\boldsymbol{v}}_i \leq \deg_{\hat{\boldsymbol{w}}} \boldsymbol{v}_i$, so we are done if we can show that $\deg \hat{v}_{i,h_i} = \deg v_{i,h_i}$ and $\deg \hat{v}_{i,k} + w_k < \deg v_{i,h_i} + w_{h_i}$ for $k > h_i$. This in turn will follow if $\alpha x^{\delta} \boldsymbol{v}_t$ has $\hat{\boldsymbol{w}}$-weighted degree less than $\deg v_{i,h_i} + w_{h_i}$ on all position $k \geq h_i$.

Due to $\mathrm{LP}_{\boldsymbol{w}}(\boldsymbol{v}_t) = h_t$ and (8) for index t then for any $k > h_t$:

$$\deg v_{t,k} + w_k < \deg v_{t,h_t} + w_{h_t} = \deg v_{t,0} + w_0 \ . \tag{9}$$

Using $\deg v_{t,0} + \delta = \deg v_{i,0}$ and (8) for index i, we conclude that

$$\deg v_{t,k} + w_k + \delta < \deg v_{i,0} + w_0 = \deg v_{i,h_i} + w_{h_i} \ .$$

Since $h_t < h_i$ by the ordering of the $i_\star$, this shows that $\deg v_{i,k} + w_k + \delta < \deg v_{i,h_i} + w_{h_i}$ for $k \geq h_i$. These are the degree bounds we sought and so $\mathrm{LP}_{\hat{\boldsymbol{w}}}(\hat{\boldsymbol{v}}_i) = h_i$.

Consider now an $i \in I$ for which Line 9 was run, and let again t be as at that point, before the reassignment. The situation is completely reversed according to before, so by analogous arguments $\mathrm{LP}_{\hat{\boldsymbol{w}}}(\hat{\boldsymbol{v}}_t) = h_i$.

For the value of t at the end of the algorithm, then clearly $\mathrm{LP}_{\hat{\boldsymbol{w}}}(\hat{\boldsymbol{v}}_t) = 0$ since the row was not modified. Since we necessarily have $h_{i_1} = 0$, then $\mathrm{LP}_{\hat{\boldsymbol{w}}}(\hat{\boldsymbol{v}}_t) = h_{i_1}$. Thus every h_i becomes the $\hat{\boldsymbol{w}}$-leading position of one of the $\boldsymbol{v}_j$ exactly once. But the h_i were all different, and so $\hat{V}$ is in $\hat{\boldsymbol{w}}$-shifted weak Popov form. □

Proposition 4 *Algorithm 6 performs at most*

$$O\big(m \deg\det(V) + \textstyle\sum_{i<j} |w_i - w_j| + m^2\big)$$

operations over $\mathcal{R}$.

Proof We will bound the number of non-zero monomials which are involved in simple transformations. As remarked in the proof of Theorem 9, all simple transformations are done using distinct rows of the input matrix, so it suffices to bound the total number of monomials in the input matrix V.

Since we are then simply counting monomials in V, we can assume w.l.o.g. that $w_0 \leq w_1 \leq \ldots \leq w_{m-1}$, and since the input matrix V was in $\boldsymbol{w}$-shifted weak Popov form, assume also w.l.o.g that we have sorted the rows such that $\mathrm{LP}_{\boldsymbol{w}}(\boldsymbol{v}_i) = i$. Since $\Delta(\Phi_{\boldsymbol{w}}(V)) = 0$ we have

$$\deg\det \Phi_{\boldsymbol{w}}(V) = \deg_{\boldsymbol{w}} V \qquad \text{that is} \qquad \deg_{\boldsymbol{w}} V = \deg\det V + \textstyle\sum_i w_i \ .$$

We can therefore consider the assignment of $\deg_{\boldsymbol{w}}$ to the individual rows of V under these constraints that will maximise the possible number of monomials in V. We cannot have $\deg_{\boldsymbol{w}} \boldsymbol{v}_i < w_i$ since $\mathrm{LP}_{\boldsymbol{w}}(\boldsymbol{v}_i) = i$. It is easy to see that the worst-case assignment is then to have exactly $\deg_{\boldsymbol{w}} \boldsymbol{v}_i = w_i$ for $i = 0, \ldots, m-2$ and $\deg_{\boldsymbol{w}} \boldsymbol{v}_{m-1} = \deg\det V + w_{m-1}$. In this case, for $i < m-1$ then $\deg v_{i,j} \leq w_i - w_j$ if $j \leq i$ and $v_{i,j} = 0$ if $j > i$, so the number of monomials can then be bounded as

$$\left(\sum_{i=0}^{m-2}\sum_{j=0}^{i}(w_i - w_j + 1)\right) + \left(\sum_{j=0}^{m-1}(\deg\det V + w_{m-1} - w_j + 1)\right)$$
$$\leq m^2 + \sum_{i<j}(w_j - w_i) + m \deg\det V. \qquad \square$$

The idea is now to iterate Algorithm 6 to "walk" from a matrix that is in weak Popov form for one shift $\boldsymbol{w}$ into another one $\hat{\boldsymbol{w}}$. Row reducing the matrix for the MV codes can be done as Algorithm 7.

Algorithm 7 Find MV interpolation polynomial by weak Popov walk

```
Input: Instance of Problem 2 and the matrix $V \leftarrow M$ of (6) on page 8
Output: A $\boldsymbol{w}$-shifted weak Popov form of $M$
1 $\boldsymbol{w} = \big(0, (k-1), 2(k-1), \ldots, \ell(k-1)\big)$.
2 $\boldsymbol{w}' = \boldsymbol{w} + \big(0, n, n, \ldots, n\big)$.
3 for $i = 0, ..., n-1$ do
4     $V \leftarrow \mathrm{WeakPopovWalk}(V, \boldsymbol{w}')$.
5     $\boldsymbol{w}' \leftarrow \boldsymbol{w}' + (1, 0, \ldots, 0)$.
6 return $V$
```

Theorem 10 *Algorithm 7 is correct. It has complexity $O(\ell n^2)$ over $\mathbb{F}_{q^s}$.*

Proof Note that M is in $\boldsymbol{w}'$-shifted weak Popov form, where $\boldsymbol{w}'$ is as on Line 2. Thus by the correctness of Algorithm 6, then V at the end of the algorithm must be in $\big(\boldsymbol{w} + (n, \ldots, n)\big)$-shifted weak Popov form. Then it is clearly also in $\boldsymbol{w}$-shifted weak Popov form. For the complexity, the algorithm simply performs n calls to Algorithm 6. We should estimate the quantity $\sum_{i<j} |w_i - w_j|$, which is greatest in the first iteration. Since Problem 2 posits $n > \binom{\ell+1}{2}(k-1)$, we can bound the sum as:

$$\sum_{j=1}^{\ell}(n + j(k-1)) + \sum_{1\le i<j}(j-i)(k-1) < \ell n + (\ell+1)\binom{\ell+1}{2}(k-1) \in O(\ell n)\ .$$

Since $\deg\det(V) = \deg\det(M) = n$ then by Proposition 4 each of the calls to Algorithm 6 therefore costs at most $O(\ell n)$. □

6 Conclusion

We have explored row reduction of skew polynomial matrices. For ordinary polynomial rings, row reduction has proven a useful strategy for obtaining flexible, efficient while conceptually simple decoding algorithms for RS and other code families. Our results introduce the methodology and tools aimed at bringing similar benefits to Gabidulin, Interleaved Gabidulin, MV, and other skew polynomial-based codes. We used those tools in two settings. We solved a general form of multiple skew-shift register synthesis (cf. Problem 1), and applied this for decoding of Interleaving Gabidulin codes in complexity $O(\ell\mu^2)$, see Theorem 2. For MV codes (cf. Problem 2), we gave an interpolation algorithm with complexity $O(\ell n^2)$, see Theorem 4.

We extended and analysed the simple and generally applicable Mulders–Storjohann algorithm to the skew polynomial setting. In both the studied settings, the complexity of that algorithm was initially not satisfactory, but it served as a crucial step in developing more efficient algorithms. For multiple skew-shift register synthesis, we were able to obtain a good complexity for a more general problem than previously. For the MV codes, the improved algorithm was in the shape of a versatile "Weak Popov Walk", which could potentially apply to many other problems. In all previously studied cases, we matched the best known complexities [44,48] that do not make use of fast multiplication of skew polynomials.

Based on a preprint of this paper, in [40] it is shown how to further reduce the complexity for decoding Interleaved Gabidulin codes using a divide-&-conquer version of Algorithm 3, matching the complexity of [43].

The weak Popov form has many properties that can be beneficial in a coding setting, and which we did not yet explore. For instance, it allows to easily enumerate all "small" elements of the row space: that could e.g. be used to enumerate *all* solutions to a shift register problem, allowing a chase-like decoding of Interleaved Gabidulin codes beyond half the minimum distance.

Acknowledgements The authors would like to thank the anonymous reviewers for suggestions that have substantially improved the clarity of the paper. Sven Puchinger (Grant BO 867/29-3), Wenhui Li and Vladimir Sidorenko (Grant BO 867/34-1) were supported by the German Research Foundation "Deutsche Forschungsgemeinschaft" (DFG).

References

1. Abramov S.A., Bronstein M.: On solutions of linear functional systems. In: Proceedings of ISSAC, pp. 1–6 (2001).
2. Alekhnovich M.: Linear Diophantine equations over polynomials and soft decoding of Reed–Solomon codes. IEEE Trans. Inf. Theory **51**(7), 2257–2265 (2005).
3. Augot D., Loidreau P., Robert G.: Rank metric and Gabidulin codes in characteristic zero. In: ISIT (2013).
4. Baker G., Graves-Morris P.: Padé Approximants, vol. 59. Cambridge University Press, Cambridge (1996).
5. Beckermann B., Labahn G.: A uniform approach for the fast computation of matrix-type Padé approximants. SIAM J. Matrix Anal. Appl. **15**(3), 804–823 (1994).
6. Beckermann B., Cheng H., Labahn G.: Fraction-free row reduction of matrices of Ore polynomials. J. Symb. Comput. **41**(5), 513–543 (2006).
7. Beelen P., Brander K.: Key equations for list decoding of Reed–Solomon codes and how to solve them. J. Symb. Comput. **45**(7), 773–786 (2010).
8. Boucher D., Ulmer F.: Linear codes using skew polynomials with automorphisms and derivations. Des. Codes Cryptogr. **70**(3), 405–431 (2014).
9. Cohn H., Heninger N.: Ideal forms of Coppersmith's theorem and Guruswami–Sudan list decoding (2010). arXiv:1008.1284.
10. Clark P.L.: Non-commutative algebra. University of Georgia. http://math.uga.edu/~pete/noncommutativealgebra.pdf (2012).
11. Delsarte P.: Bilinear forms over a finite field, with applications to coding theory. J. Comb. Theory **25**(3), 226–241 (1978).
12. Dieudonné J.: Les déterminants sur un corps non commutatif. Bull. Soc. Math. France **71**, 27–45 (1943).
13. Draxl P.K.: Skew Fields, vol. 81. Cambridge University Press, Cambridge (1983).
14. Feng G.L., Tzeng K.K.: A generalization of the Berlekamp–Massey algorithm for multisequence shift-register synthesis with applications to decoding cyclic codes. IEEE Trans. Inf. Theory **37**(5), 1274–1287 (1991).
15. Gabidulin E.M.: Theory of codes with maximum rank distance. Problemy Peredachi Informatsii **21**(1), 3–16 (1985).
16. Giorgi P., Jeannerod C., Villard G.: On the complexity of polynomial matrix computations. In: Proceedings of ISSAC, pp. 135–142 (2003).
17. Guruswami V., Sudan M.: Improved decoding of Reed–Solomon codes and algebraic-geometric codes. IEEE Trans. Inf. Theory **45**(6), 1757–1767 (1999).
18. Guruswami V., Wang C.: Explicit rank-metric codes list-decodable with optimal redundancy. In: Proceedings of RANDOM (2014). arXiv:1311.7084.
19. Guruswami V., Xing C.: List decoding Reed-solomon, algebraic-geometric, and Gabidulin subcodes up to the singleton bound. In: Proceedings of STOC, pp. 843–852. ACM, New York (2013)
20. Kailath T.: Linear Systems. Prentice-Hall, Upper Saddle River (1980).
21. Lee K., O'Sullivan M.E.: List decoding of Reed–Solomon codes from a Gröbner basis perspective. J. Symb. Comput. **43**(9), 645–658 (2008).
22. Lenstra A.: Factoring multivariate polynomials over finite fields. J. Comput. Syst. Sci. **30**(2), 235–246 (1985).
23. Li W., Sidorenko V., Silva D.: On transform-domain error and erasure correction by Gabidulin codes. Des. Codes Cryptogr. **73**(2), 571–586 (2014).
24. Li W., Nielsen J.S.R., Puchinger S., Sidorenko V.: Solving shift register problems over skew polynomial rings using module minimisation. In: Proceedings of WCC (2015).
25. Loidreau P., Overbeck R.: Decoding rank errors beyond the error correcting capability. In: Proceedings of ACCT, pp. 186–190 (2006).
26. Mahdavifar H.: List decoding of subspace codes and rank-metric codes. Ph.D. thesis, University of California, San Diego (2012).
27. Mahdavifar H., Vardy A.: Algebraic list-decoding of subspace codes with multiplicities. In: Allerton Conference on Communication, Control, and Computing, pp. 1430–1437 (2011).
28. Mahdavifar H., Vardy A.: Algebraic list-decoding of subspace codes. IEEE Trans. Inf. Theory **59**(12), 7814–7828 (2013).
29. Middeke J.: A computational view on normal forms of matrices of Ore polynomials. Ph.D. thesis, Research Institute for Symbolic Computation (RISC) (2011).
30. Müelich S., Puchinger S., Mödinger D., Bossert M.: An alternative decoding method for Gabidulin codes in characteristic zero. In: Proceedings of IEEE ISIT (2016). arXiv:1601.05205.
31. Mulders T., Storjohann A.: On lattice reduction for polynomial matrices. J. Symb. Comput. **35**(4), 377–401 (2003).

32. Nielsen J.S.R.: Generalised multi-sequence shift-register synthesis using module minimisation. In: Proceedings of IEEE ISIT, pp. 882–886 (2013).
33. Nielsen J.S.R.: Power decoding Reed–Solomon codes up to the Johnson radius. In: Proceedings of ACCT (2014).
34. Nielsen J.S.R., Beelen P.: Sub-quadratic decoding of one-point Hermitian codes. IEEE Trans. Inf. Theory **61**(6), 3225–3240 (2015).
35. Nielsen R.R., Høholdt T.: Decoding Reed–Solomon codes beyond half the minimum distance. In: Coding Theory, Cryptography and Related Areas, pp. 221–236. Springer, Berlin (1998).
36. Olesh Z., Storjohann A.: The vector rational function reconstruction problem. In: Proceedings of WWCA, pp. 137–149 (2006).
37. Ore O.: On a special class of polynomials. Trans. Am. Math. Soc. **35**(3), 559–584 (1933).
38. Ore O.: Theory of non-commutative polynomials. Ann. Math. **34**(3), 480–508 (1933).
39. Puchinger S., Wachter-Zeh A.: Fast operations on linearized polynomials and their applications in coding theory. J. Symb. Comput. (2015). arXiv:1512.06520.
40. Puchinger S., Müelich S., Mödinger D., Nielsen J.S.R., Bossert M.: Decoding interleaved Gabidulin codes using Alekhnovich's algorithm. In: Proceedings of ACCT (2016). arXiv:1604.04397.
41. Roth R.M.: Maximum-rank array codes and their application to crisscross error correction. IEEE Trans. Inf. Theory **37**(2), 328–336 (1991).
42. Roth R., Ruckenstein G.: Efficient decoding of Reed–Solomon codes beyond half the minimum distance. IEEE Trans. Inf. Theory **46**(1), 246–257 (2000).
43. Sidorenko V., Bossert M.: Fast skew-feedback shift-register synthesis. Des. Codes Cryptogr. **70**(1–2), 55–67 (2014).
44. Sidorenko V., Jiang L., Bossert M.: Skew-feedback shift-register synthesis and decoding interleaved Gabidulin codes. IEEE Trans. Inf. Theory **57**(2), 621–632 (2011).
45. Sidorenko V., Schmidt G.: A linear algebraic approach to multisequence shift-register synthesis. Probl. Inf. Transm. **47**(2), 149–165 (2011).
46. Taelman L.: Dieudonné determinants for skew polynomial rings. J. Algebra Appl. **5**(1), 89–93 (2006).
47. Wachter-Zeh A.: Decoding of block and convolutional codes in rank metric. Ph.D. thesis, Universität Ulm (2013).
48. Xie H., Lin J., Yan Z., Suter B.W.: Linearized polynomial interpolation and its applications. IEEE Trans. Signal Process. **61**(1), 206–217 (2013).
49. Zeh A., Gentner C., Augot D.: An interpolation procedure for list decoding Reed–Solomon codes based on generalized key equations. IEEE Trans. Inf. Theory **57**(9), 5946–5959 (2011).
50. Zhou W., Labahn G.: Efficient algorithms for order basis computation. J. Symb. Comput. **47**(7), 793–819 (2012).