

# Short proofs of ideal membership

Clemens Hofstadler<sup>a</sup>, Thibaut Verron<sup>b</sup>

<sup>a</sup>*Institute of Mathematics, University of Kassel, Heinrich-Plett-Straße 40, Kassel, Germany*

<sup>b</sup>*Institute for Algebra, Johannes Kepler University, Altenberger Straße 69, Linz, Austria*

---

## Abstract

A cofactor representation of an ideal element, that is, a representation in terms of the generators, can be considered as a certificate for ideal membership. Such a representation is typically not unique, and some can be a lot more complicated than others. In this work, we consider the problem of computing sparsest cofactor representations, i.e., representations with a minimal number of terms, of a given element in a polynomial ideal. While we focus on the more general case of noncommutative polynomials, all results also apply to the commutative setting.

We show that the problem of computing cofactor representations with a bounded number of terms is decidable and NP-complete. Moreover, we provide a practical algorithm for computing sparse (not necessarily optimal) representations by translating the problem into a linear optimization problem and by exploiting properties of signature-based Gröbner basis algorithms. We show that, for a certain class of ideals, representations computed by this method are actually optimal, and we present experimental data illustrating that it can lead to noticeably sparser cofactor representations.

*Keywords:* Noncommutative polynomials, Signature Gröbner basis, Automated proofs, Proof simplification, Linear programming

---

## 1. Introduction

In polynomial algebra, the ideal membership problem is one of the most fundamental problems with many important applications from polynomial system solving over polynomial identity testing to automated reasoning. In case of noncommutative polynomials in the free algebra, this last application is particularly relevant and has been the focus of recent research [HW94, SL20, CHRR20, RRHP21, BHR23, Hof23]. Several frameworks have been developed that allow to prove the correctness of statements about linear operators (such as matrices, homomorphisms, bounded operators, etc.) by verifying ideal membership of noncommutative polynomials. In this setting, any cofactor representation, that is, any representation of an ideal element as a linear combination of the generators, can be considered as a proof of the corresponding operator statement, and

---

*Email addresses:* clemens.hofstadler@mathematik.uni-kassel.de (Clemens Hofstadler), thibaut.verron@gmail.com (Thibaut Verron)

noncommutative Gröbner bases can be used to compute such representations [HRR19], see also [LSAZ20] and references therein for available software.

In general, cofactor representations are not unique and different representations can differ drastically in their complexity. We could observe empirically that representations computed by Gröbner bases are often significantly longer than necessary. In this work, we discuss the problem of finding *sparsest* cofactor representations of an ideal element, that is, representations with a minimal number of terms. We focus on the situation of noncommutative polynomials, as we are particularly interested in computing short proofs of operator statements. However, all techniques also apply analogously to commutative polynomials.

Although ideal membership in the free algebra is only semidecidable, we show that the problem of computing cofactor representations with the number of terms bounded by  $N \in \mathbb{N}$  is decidable, yet NP-complete. This yields a first (impractical) algorithm for computing sparsest cofactor representations (Algorithm 1).

We then describe how to obtain a practical algorithm for computing sparse (not necessarily sparsest) representations by making two simplifications:

1. We restrict the search space to a finite dimensional subspace by only considering cofactor representations with terms smaller than a designated bound.
2. We use the sum of the absolute values of the coefficients, i.e., the  $\ell_1$ -norm, of a representation as a complexity measure.

With these simplifications, we translate the problem of finding sparse cofactor representations into solving a linear programming problem. Our main result is Algorithm 3, which computes, starting from any given cofactor representation, a minimal one w.r.t. the conditions (1) and (2).

We also show that the second simplification does in fact impose no restriction for a class of ideals that appears frequently when translating operator statements. In particular, we prove that, under certain assumptions satisfied by most examples studied in practice, Algorithm 3 computes a sparsest representation among all representations satisfying condition (1). Finally, we demonstrate the effectiveness of Algorithm 3 on several examples coming from actual operator statements.

Our algorithm relies on information provided by noncommutative signature-based Gröbner basis algorithms. Initially developed in the commutative setting to improve Gröbner basis computations, signature-based algorithms have been subject of extensive research, see [EF17] and [Lai22]. Recently, the authors proposed a generalization of these algorithms to free algebras over fields [HV22] and rings [HV23].

Signature-based algorithms compute, in addition to a Gröbner basis, some information on how the polynomials in that basis were computed. This additional information allows the algorithms not only to predict and avoid redundant computations, but also to compute a Gröbner basis of the syzygy module of the generators. In particular, signature-based algorithms compute this basis in a very structured way, and precisely this structure is what we exploit in our algorithm.

To the best of our knowledge, the problem at hand has not been studied in this form, that is, for noncommutative polynomials and from a practical point of view. For commutative polynomials, the complexity of the ideal membership problem has been

the focus of extensive research. Starting from early work showing that the problem requires, in general, doubly exponential time [MM82, May89], to recent results identifying special classes of polynomial ideals, often encompassing systems arising in applications in cryptography, physics or combinatorics, for which it can be solved more efficiently [FS12, FSS13, FSS14, FSEDV16, MT17, Mas21, BR22b, BR22a]. The (commutative) ideal membership problem is also central to the field of algebraic proof complexity, where finding sparse cofactor representations corresponds to finding *short Nullstellensatz proofs*, see [PT16] for a recent survey. In this setting, plenty of work is dedicated to establishing degree bounds of commutative cofactor representations in the form of effective versions of Hilbert’s Nullstellensatz [Kol88, Jel05, dKS13]. It is worth noting that the research listed above primarily concentrates on theoretical aspects without considering the practical applicability of the developed methods and algorithms. In contrast, in this work, we focus on actual computability of sparse representations for concrete examples. Proof simplification in automated theorem proving in general is also addressed in e.g., [Ver01, Kin19].

## 2. Preliminaries

We recall basic definitions regarding noncommutative polynomials and signature-based Gröbner basis algorithms in the free algebra. For more details, we refer to [HV22, HV23]. For an introduction to noncommutative Gröbner bases without signatures, see [Xiu12, Mor16].

### 2.1. Free algebra

Throughout this paper,  $X = \{x_1, \dots, x_k\}$  is a finite set of indeterminates and  $\langle X \rangle$  is the free monoid over  $X$ . For  $m \in \langle X \rangle$ , we denote by  $|m|$  the length of  $m$ . Let  $K$  be a field and  $K\langle X \rangle$  be the free algebra over  $X$ . We consider the elements in  $K\langle X \rangle$  as *noncommutative polynomials*, where monomials are given by words over  $X$  and multiplication is given by concatenation of words. Note that  $K\langle X \rangle$  is not Noetherian if  $|X| > 1$ . For  $F \subseteq K\langle X \rangle$ , we denote by  $\langle F \rangle$  the (two-sided) ideal generated by  $F$ .

A *monomial ordering* on  $\langle X \rangle$  is a well-ordering  $\leq$  compatible with the multiplication in  $\langle X \rangle$ , that is,  $m \leq m'$  implies  $amb \leq am'b$  for all  $a, b, m, m' \in \langle X \rangle$ . We fix a monomial ordering  $\leq$ .

*Example 2.1.* The degree lexicographic ordering  $\leq_{\text{deglex}}$  is a monomial ordering on  $\langle X \rangle$  where two words  $m, m' \in \langle X \rangle$  are first compared by their lengths and ties are broken by comparing the variables in  $m$  and  $m'$  from left to right using the lexicographic ordering  $x_1 <_{\text{lex}} \dots <_{\text{lex}} x_k$ .

For  $f \in K\langle X \rangle$ , the support of  $f$ , denoted by  $\text{supp}(f)$ , is the set of all monomials appearing in  $f$ . For  $f \neq 0$ , the leading monomial  $\text{lm}(f)$  of  $f$  is the maximal element w.r.t.  $\leq$  in  $\text{supp}(f)$  and the degree  $\text{deg}(f)$  of  $f$  is  $\text{deg}(f) = \max_{m \in \text{supp}(f)} |m|$ . Additionally, we set  $\text{deg}(0) = -1$ .

## 2.2. Free bimodule

We fix a family of polynomials  $f_1, \dots, f_r \in K\langle X \rangle$ , generating an ideal  $I = (f_1, \dots, f_r)$ . We extend the previous definitions to the *free*  $K\langle X \rangle$ -bimodule  $\Sigma$  of rank  $r \in \mathbb{N}$ , given by  $\Sigma = (K\langle X \rangle \otimes K\langle X \rangle)^r$ . The canonical basis of  $\Sigma$  is  $\varepsilon_1, \dots, \varepsilon_r$ , where  $\varepsilon_i = (0, \dots, 0, 1 \otimes 1, 0, \dots, 0)$  with  $1 \otimes 1$  appearing in the  $i$ th position for  $i = 1, \dots, r$ .

The set  $M(\Sigma)$  of *module monomials* in  $\Sigma$  is given by  $M(\Sigma) = \{a\varepsilon_i b \mid a, b \in \langle X \rangle, i = 1, \dots, r\}$ . Every element  $\alpha \in \Sigma$  has a unique representation of the form  $\alpha = \sum_{i=1}^d c_i a_i \varepsilon_{j_i} b_i$  with nonzero  $c_i \in K$  and pairwise different  $a_i \varepsilon_{j_i} b_i \in M(\Sigma)$ . We denote its support by  $\text{supp}(\alpha) = \{a_i \varepsilon_{j_i} b_i \mid i = 1, \dots, d\}$  and associate to it the polynomial  $\bar{\alpha} := \sum_{i=1}^d c_i a_i f_{j_i} b_i \in I$ . With this, the (weighted) degree of  $\alpha$  is  $\deg(\alpha) = \max_i \deg(a_i f_{j_i} b_i)$ .

A *module ordering* on  $M(\Sigma)$  is a well-ordering  $\leq$  compatible with scalar multiplication, that is,  $\mu \leq \mu'$  implies  $a\mu b \leq a\mu' b$  for all  $\mu, \mu' \in M(\Sigma)$  and  $a, b \in \langle X \rangle$ . We fix a module ordering  $\leq$ .

*Example 2.2.* The degree-over-position-over-term ordering  $\leq_{\text{DoPoT}}$  is a module ordering where two module monomials  $a\varepsilon_i b, a'\varepsilon_j b' \in M(\Sigma)$  are first compared by their degrees and ties are broken by comparing the tuples  $(i, a, b)$  and  $(j, a', b')$  lexicographically using a monomial ordering for the monomial comparisons.

**Definition 2.3.** The *signature*  $\mathfrak{s}(\alpha)$  of a nonzero  $\alpha \in \Sigma$  is the maximal element w.r.t.  $\leq$  in  $\text{supp}(\alpha)$ .

## 2.3. Signature-based algorithms

For the rest of this work, we assume that the monomial ordering  $\leq$  and the module ordering  $\leq$  satisfy:

- $\leq$  and  $\leq$  are *compatible* in the sense that  $a < b$  iff  $a\varepsilon_i < b\varepsilon_i$  iff  $\varepsilon_i a < \varepsilon_i b$  for all  $a, b \in \langle X \rangle$  and  $i = 1, \dots, r$ ;
- $\leq$  is *fair*, meaning that the set  $\{\mu' \in M(\Sigma) \mid \mu' < \mu\}$  is finite for all  $\mu \in M(\Sigma)$ ;

*Example 2.4.* The module ordering  $\leq_{\text{DoPoT}}$  is fair and compatible with  $\leq_{\text{deglex}}$ .

A *labeled polynomial*  $f^{[\alpha]}$  is a pair  $(f, \alpha) \in I \times \Sigma$  with  $f = \bar{\alpha}$ . We call the set  $I^{[\Sigma]} := \{f^{[\alpha]} \mid f \in I, \bar{\alpha} = f\}$  the *labeled module* generated by  $f_1, \dots, f_r$ . It forms a  $K\langle X \rangle$ -subbimodule of  $I \times \Sigma$  with component-wise addition and scalar multiplication.

A syzygy of  $I^{[\Sigma]}$  is an element  $\gamma \in \Sigma$  such that  $\bar{\gamma} = 0$ . The set of syzygies of  $I^{[\Sigma]}$ , denoted by  $\text{Syz}(I^{[\Sigma]})$ , forms a  $K\langle X \rangle$ -subbimodule of  $\Sigma$ . We recall the notion of Gröbner basis of  $\text{Syz}(I^{[\Sigma]})$ .

**Definition 2.5.** A set  $H \subseteq \text{Syz}(I^{[\Sigma]})$  is a *Gröbner basis* of  $\text{Syz}(I^{[\Sigma]})$  (up to signature  $\sigma \in M(\Sigma)$ ) if for all nonzero  $\gamma \in \text{Syz}(I^{[\Sigma]})$  (with  $\mathfrak{s}(\gamma) < \sigma$ ), there exist  $d \in \mathbb{N}$  and  $\gamma_i \in H$ ,  $c_i \in K$ ,  $a_i, b_i \in \langle X \rangle$  such that  $\gamma = \sum_{i=1}^d c_i a_i \gamma_i b_i$  and  $\mathfrak{s}(a_i \gamma_i b_i) \leq \mathfrak{s}(\gamma)$  for all  $i = 1, \dots, d$ .

In general, a syzygy module need not have a finite Gröbner basis. Nevertheless, Gröbner bases of  $\text{Syz}(I^{[\Sigma]})$  can be enumerated by increasing signatures using signature-based algorithms.

**Theorem 2.6.** *There exists an algorithm to correctly enumerate a Gröbner basis of  $\text{Syz}(I^{[\Sigma]})$  in increasing signature order. In particular, for all  $\sigma \in M(\Sigma)$ , stopping the algorithm at the first syzygy with signature  $\geq \sigma$  yields a finite Gröbner basis of  $\text{Syz}(I^{[\Sigma]})$  up to signature  $\sigma$ .*

*Proof.* The algorithm is [HV22, Algo. 1]. Its correctness is proved in [HV22, Thm. 42] and the fact that the signatures increase throughout is proved in [HV22, Lem. 43].  $\square$

The algorithm uses signature-based Gröbner techniques, and also enumerates a (possibly infinite) *labeled Gröbner basis* of  $I^{[\Sigma]}$ . The definition and construction of labeled Gröbner bases are beyond the scope of this paper, but we note that, among their properties, a labeled Gröbner basis is a set  $G^{[\Sigma]} \subseteq I^{[\Sigma]}$  such that the set  $\{f \mid f^{[\alpha]} \in G^{[\Sigma]}\}$  is a Gröbner basis of  $I$ . In particular, given  $f \in I$ , performing polynomial reductions by  $G^{[\Sigma]}$  to reduce  $f$  to 0 and adding the labeling of the reducers yields a cofactor representation of  $f$  w.r.t.  $f_1, \dots, f_r$ .

*Remark 2.7.* The algorithm mentioned in the proof of Theorem 2.6 is inefficient as it has to perform a lot of expensive module arithmetic. A more efficient way of realizing Theorem 2.6 is to combine [HV22, Algo. 2.4] for computing *signature Gröbner bases*. These algorithms work with pairs  $(f, \mathfrak{s}(\alpha))$  instead of labeled polynomials  $f^{[\alpha]}$  and reconstruct the full module representations *a posteriori*, avoiding a lot of module arithmetic in this way.

A finite Gröbner basis of  $\text{Syz}(I^{[\Sigma]})$  up to signature  $\sigma$ , as output by the signature-based algorithm, provides an effective description of all syzygies with signature  $< \sigma$ . This is the crucial property of signature-based algorithms that we exploit in Section 4 to compute sparse cofactor representations.

### 3. Decidability and complexity

For the rest of this section, we fix a family of polynomials  $f_1, \dots, f_r \in K\langle X \rangle$  generating an ideal  $I$ . Any cofactor representation of an ideal member  $f \in I$  can be identified with an element  $\alpha \in \Sigma$  such that  $\bar{\alpha} = f$ . The *weight* of such a representation is given by the  $\ell_0$ -“norm”  $\|\alpha\|_0 := |\text{supp}(\alpha)|$ . Then, with the set  $R(f) := \{\alpha \in \Sigma \mid \bar{\alpha} = f\}$  of (cofactor) representations of  $f$ , a *sparsest* (cofactor) representation of  $f$  corresponds to a minimal element w.r.t.  $\|\cdot\|_0$  in  $R(f)$ . We denote the set of all such minimal elements by  $R_0(f)$ . If  $f \notin I$ , we set  $R(f) = R_0(f) = \emptyset$ .

*Remark 3.1.* Ideal membership in the free algebra is only semidecidable. Consequently, we can also not decide whether  $R(f) = \emptyset$  or not.

*Remark 3.2.* The function  $\|\cdot\|_0$  is not a norm as it is not homogeneous, but one can associate to it a metric called *Hamming distance*.

In the following, we study the decidability and complexity of computing cofactor representations of bounded weight. To this end, we assume that the coefficient field  $K$  is computable, in the sense that the basic arithmetic operations as well as equality testing are effective. This means in particular that linear systems can be solved effectively using, for example, Gaussian elimination. With this, we consider the following problem.

**Problem 3.3** (Sparse cofactor representation).

INPUT:  $f, f_1, \dots, f_r \in K\langle X \rangle, N \in \mathbb{N}$

OUTPUT: a cofactor representation  $\alpha \in R(f)$  with  $\|\alpha\|_0 \leq N$  if one exists, otherwise False.

We show that Problem 3.3 is decidable, and we give an algorithm reducing it to Problem 3.4 below of finding sparse solutions of linear systems, formally also known as the Min-RVLS (MINimum Relevant Variables in Linear System) problem. This not only yields an algorithm for computing sparsest cofactor representations if  $f \in I$ , but it, in principle, also provides a semidecision procedure for ideal membership. We focus on the first application here.

**Problem 3.4** (Sparse solution of linear system [Min-RVLS]).

INPUT:  $A \in K^{m \times n}, \mathbf{b} \in K^m, N \in \{0, \dots, n\}$

OUTPUT: a vector  $\mathbf{y} \in K^n$  with  $A\mathbf{y} = \mathbf{b}$  and  $\|\mathbf{y}\|_0 \leq N$  if one exists, otherwise False.

Problem 3.4 arises in many areas [CDS01, CT05, Don06]. It is clearly decidable, for instance by looping over all  $N$ -subsets of  $\{1, \dots, n\}$  for possible sets of nonzero coefficients of solutions. Furthermore, in many cases, most notably for  $K = \mathbb{Q}$ , it is known to be NP-hard, with the corresponding decision problem being NP-complete [GJ79, Problem MP5].

In order to reduce the sparse cofactor representation problem to linear algebra, we need to constrain the solutions to a finite dimensional vector space, which requires to bound the degree of a solution. The degree of elements in  $R(f)$  can be arbitrarily large, but we can bound the degree of *minimal* representations. A cofactor representation  $\alpha = \sum_{i=1}^d c_i a_i \varepsilon_{j_i} b_i \in R(f)$  is *minimal* if no sub-sum is a syzygy, that is,  $\sum_{i \in J} c_i a_i \varepsilon_{j_i} b_i \neq 0$  for all non-empty subsets  $J \subseteq \{1, \dots, d\}$ . To obtain the degree bound, we recall the notion of (*polynomial*) *rewriting* introduced in [RRHP21, Def. 2].

**Definition 3.5.** Let  $f, g \in K\langle X \rangle$  and  $a, b \in \langle X \rangle$  such that  $\text{supp}(f) \cap \text{supp}(agb) \neq \emptyset$ . For every  $c \in K$ , we say that  $f$  can be rewritten to  $f + cagb \in K\langle X \rangle$  by  $g$ .

Furthermore, we say that  $f$  can be rewritten to  $h$  by  $G \subseteq K\langle X \rangle$  if there are  $h_0, \dots, h_d \in K\langle X \rangle, h_d = f, h_0 = h$  and  $g_1, \dots, g_d \in G$  such that  $h_k$  can be rewritten to  $h_{k-1}$  by  $g_k$  for all  $k = 1, \dots, d$ .

Rewriting can be considered as a weaker form of polynomial reduction, not requiring that a polynomial gets “simplified” by a rewriting step. Nevertheless,  $f$  can be rewritten to zero by  $\{f_1, \dots, f_r\}$  if and only if  $f \in I$ , see [RRHP21, Lem. 4]. More importantly, we can show that any minimal representation of  $f$  can be obtained by rewriting  $f$  to 0 by  $\{f_1, \dots, f_r\}$  and logging the rewriting steps.

**Lemma 3.6.** Let  $f \in (f_1, \dots, f_r)$  and  $\alpha = \sum_{i=1}^d c_i a_i \varepsilon_{j_i} b_i \in R(f)$  be a minimal representation of  $f$ . Furthermore, for  $k = 0, \dots, d$ , let  $h_k = \sum_{i=1}^k c_i a_i \varepsilon_{j_i} b_i$ . In particular,  $h_d = f$  and  $h_0 = 0$ . Then, possibly after reordering the summands of  $\alpha$ ,  $h_k$  can be rewritten to  $h_{k-1}$  by  $\{f_1, \dots, f_r\}$  for all  $k = 1, \dots, d$ .

*Proof.* We perform induction on the weight  $d$  of a minimal representation of  $f$ . For  $d = 0$  there is nothing to prove. Assume now that  $d > 0$  and that the result is proven

for polynomials with a minimal representation of weight  $d - 1$ . Because  $\alpha$  is a minimal representation,  $f$  cannot be 0. Since  $f = \bar{\alpha} = \sum_{i=1}^d c_i a_i f_{j_i} b_i$ , the support of  $f$  is contained in the union of the supports of the  $a_i f_{j_i} b_i$ , and there exists  $1 \leq k \leq d$  such that  $\text{supp}(f) \cap \text{supp}(a_k f_{j_k} b_k) \neq \emptyset$ . Possibly after reordering the summands of  $\alpha$ , we can assume  $k = d$ . So  $f$  can be rewritten to  $h_{d-1} = f - c_d a_d f_{j_d} b_d$  using  $f_{j_d} \in \{f_1, \dots, f_r\}$ . Furthermore,  $h_{d-1} = \sum_{i=1}^{d-1} c_i a_i f_{j_i} b_i$  has a minimal representation of weight  $d - 1$ , because adding one term results in a minimal representation of weight  $d$  for  $f$ . So, by induction hypothesis, this representation of  $h_{d-1}$  is (up to reordering of the summands) a sequence of rewritings by  $\{f_1, \dots, f_r\}$ .  $\square$

Another crucial property of rewriting is that we can bound the degree of the output in terms of the degree of the input and the *degree difference* of the rewriter. The *degree difference*  $\text{degdiff}(g)$  of a nonzero  $g \in K\langle X \rangle$  is  $\text{degdiff}(g) = \text{deg}(g) - \text{degmin}(g)$ , where  $\text{degmin}(g) = \min_{m \in \text{supp}(g)} |m|$ .

**Lemma 3.7.** *Let  $f, g \in K\langle X \rangle$  and  $c \in K, a, b \in \langle X \rangle$ . If  $f$  can be rewritten to  $h = f + cagb$  by  $g$ , then  $\max\{\text{deg}(h), \text{deg}(agb)\} \leq \text{deg}(f) + \text{degdiff}(g)$ .*

*Proof.* By definition, there exists a monomial  $m$  in  $\text{supp}(g)$  such that  $amb \in \text{supp}(f)$ , so  $|amb| \leq \text{deg}(f)$ , or equivalently  $|a| + |b| \leq \text{deg}(f) - |m|$ . Since  $m \in \text{supp}(g)$ ,  $|m| \geq \text{degmin}(g)$ , and all in all,

$$\begin{aligned} \text{deg}(agb) &= |a| + |b| + \text{deg}(g) \\ &\leq \text{deg}(f) - \text{degmin}(g) + \text{deg}(g) = \text{deg}(f) + \text{degdiff}(g). \end{aligned}$$

As  $\text{deg}(h) \leq \max\{\text{deg}(f), \text{deg}(agb)\}$  and  $\text{degdiff}(g) \geq 0$ , we conclude that also  $\text{deg}(h) \leq \text{deg}(f) + \text{degdiff}(g)$ .  $\square$

Combining Lemma 3.6 and 3.7, we obtain a bound on the degree of minimal cofactor representations of  $f$  of bounded weight. Since any sparsest cofactor representation is, in particular, minimal, this also yields a bound on the degree of sparsest representations.

**Corollary 3.8.** *Let  $f \in (f_1, \dots, f_r)$ ,  $N \in \mathbb{N}$  and  $\alpha \in R(f)$  be a minimal representation of  $f$ . If  $\|\alpha\|_0 \leq N$ , then  $\text{deg}(\alpha) \leq \text{deg}(f) + N \max_i \text{degdiff}(f_i)$ .*

*Proof.* Write  $\alpha$  as  $\alpha = \sum_{i=1}^d c_i a_i \varepsilon_{j_i} b_i$ . By definition,  $\text{deg}(\alpha) = \max_i \text{deg}(a_i f_{j_i} b_i)$ , and, according to Lemma 3.6, each  $a_i f_{j_i} b_i$  is a rewriter in a rewriting sequence from  $f$  to 0. Thus, Lemma 3.7 shows inductively that  $\text{deg}(\alpha) \leq \text{deg}(f) + \|\alpha\|_0 \max_i \text{degdiff}(f_i)$  and the result follows since  $\|\alpha\|_0 \leq N$ .  $\square$

As a consequence, we can state an algorithm for computing a cofactor representation of weight bounded by  $N \in \mathbb{N}$ , reducing to the problem of finding a sparse solution of a linear system.

**Corollary 3.9.** *Algorithm 1 terminates and is correct.*

*Proof.* The algorithm reduces the problem to that of finding sparse solutions of a linear system. This problem is decidable (recall that  $K$  is computable), so the algorithm terminates.

---

**Algorithm 1:** Sparse cofactor representation

---

**Input:**  $f, f_1, \dots, f_r \in K\langle X \rangle, N \in \mathbb{N}$

**Output:**  $\alpha \in R(f)$  with  $\|\alpha\|_0 \leq N$  if one exists, otherwise **False**

- 1  $D \leftarrow \deg(f) + N \max_i \deg \text{diff}(f_i)$  ;
  - 2  $L \leftarrow \{af_i b \mid a, b \in \langle X \rangle, \deg(af_i b) \leq D, i = 1, \dots, r\}$  ;
  - 3 **return** a  $K$ -linear combination of elements of  $L$  equal to  $f$  with  $\leq N$  nonzero summands if one exists, otherwise **False**;
- 

There exists a representation of  $f$  of weight  $\leq N$  if and only if there exists a *minimal* representation  $\alpha$  of  $f$  of weight  $\leq N$ . By Corollary 3.8, this representation is given by a linear combination of weight  $\|\alpha\|_0 \leq N$  consisting of elements of  $L$ . So the algorithm is correct.  $\square$

It is also possible to describe a reduction of Problem 3.4 to Problem 3.3, which allows us to characterize the complexity of the problem of finding sparse representations in terms of the complexity of Problem 3.4. For the practically most relevant case of  $K = \mathbb{Q}$  we arrive at the following theorem.

**Theorem 3.10.** *The problem of, given  $f, f_1, \dots, f_r \in \mathbb{Q}\langle X \rangle$  and  $N \in \mathbb{N}$  (in unary form), deciding whether there exists a cofactor representation of  $f$  of weight at most  $N$ , is NP-complete.*

*Proof.* Over  $\mathbb{Q}$ , the decision problem associated to Problem 3.4 is NP-complete [GJ79, Problem MP5]. Given an input  $A, \mathbf{b}, N$  to that problem, introduce one variable  $x_i$  for each row of  $A$ , interpret each column of  $A$  as the polynomial  $f_j = \sum_i A_{i,j} x_i$ , and the right-hand side as the polynomial  $f = \sum_i \mathbf{b}_i x_i$ . There is a one-to-one correspondence between solutions with  $N$  nonzero entries to the linear system and cofactor representations of  $f$  with weight  $N$ . So the problem of finding a representation of weight at most  $N$  is also NP-hard.

Furthermore, if there exists a representation of weight  $\leq N$ , then there exists one with degree  $\leq \deg(f) + N \max_i \deg \text{diff}(f_i)$ , which makes it polynomial size in  $N$  and the size of the input polynomials. The validity of that representation can be verified in polynomial time. So the problem is NP, and therefore NP-complete.  $\square$

*Remark 3.11.* The requirement that  $N$  be given in unary format is necessary in order to get a bound in terms of the size of the input: unlike in Problem 3.4, the input of Problem 3.3 is not at least of size  $N$ . If  $N$  is given in binary format, the size of the input is  $\log(N)$  + the length of a representation of the polynomials, and the latter is independent of  $N$ . In that case, the decision problem is still NP-hard but no longer NP, because the degree bound is not polynomial in  $\log(N)$ .

Since Problem 3.4 is NP-complete, Theorem 3.10 implies that there exists a polynomial-time reduction of Problem 3.3 to Problem 3.4. However, Algorithm 1 is not such a reduction, as it reduces to an instance of Problem 3.4 with size  $|L| = O(k^N)$ , where  $k$  is the number of variables. For Problem 3.4, the complexity of a brute-force solver looping over all  $N$ -subsets of  $\{1, \dots, n\}$  is  $O(n^N)$ . All in all, the runtime complexity of Algorithm 1, using such an algorithm for the last step, is  $O(k^{N^2})$ .



In practice, the last step of Algorithm 1 is infeasible for non-trivial examples. To illustrate this point, we consider the following simple statement about the Moore-Penrose inverse.

**Theorem 3.12** ([Hog13, Ch. 5.7 Fact 11]). *Let  $A$  be an invertible matrix with inverse  $B$ . Then  $B$  is the Moore-Penrose inverse of  $A$ .*

*Proof.* Let  $A^\dagger$  be the Moore-Penrose inverse of  $A$ . So in particular,  $AA^\dagger A = A$ , and hence  $B = BAB = BAA^\dagger AB = BAA^\dagger = A^\dagger$ .  $\square$

*Example 3.13.* Theorem 3.12 can be encoded in terms of the ideal membership  $b - a^\dagger \in (F)$  with

$$F = \{ab - 1, ba - 1, aa^\dagger a - a, a^\dagger aa^\dagger - a^\dagger, (a^\dagger)^* a^* - aa^\dagger, a^*(a^\dagger)^* - a^\dagger a\}$$

in the algebra  $\mathbb{Q}\langle a, a^*, a^\dagger, (a^\dagger)^*, b \rangle$ .

The proof given above is then equivalent to the following cofactor representation of  $b - a^\dagger$  certifying the ideal membership:

$$b - a^\dagger = a^\dagger(ab - 1) - b(ab - 1) - b(aa^\dagger a - a)b + (ba - 1)a^\dagger ab. \quad (1)$$

This cofactor representation consists of 4 terms. To see if there exists a representation with  $\leq 3$  terms, we can call Algorithm 1 with  $N = 3$ . The set  $L$  contains polynomials of degree at most  $D = 7$ , and it consists of 88 672 elements. This is too large to test all 3-subsets exhaustively.

Using the techniques of Section 4, we will see that a much smaller set of elements is sufficient and by applying the results of Section 4.5, we will be able to verify that (1) is in fact a sparsest representation of  $b - a^\dagger$ . This shows that the proof given above is a shortest proof of the theorem.

#### 4. Computing sparse representations

For the rest of this work, we restrict ourselves to the case  $K = \mathbb{Q}$ . We have seen in the previous section that computing sparsest cofactor representations is equivalent to the NP-hard problem of finding sparsest solutions of a linear system. Several methods have been proposed to obtain approximate solutions of the latter [CW92, MZ93, CDS01] by using other measures as proxies for the sparsity of a solution and by minimizing over them. One of these methods, called *Basis Pursuit* [CDS01], uses the  $\ell_1$ -norm as an approximation for the sparsity of a solution.

In the following, we follow the Basis Pursuit approach and use the  $\ell_1$ -norm  $\|\alpha\|_1 := \sum_{i=1}^d |c_i|$  of  $\alpha = \sum_{i=1}^d c_i a_i \varepsilon_{j_i} b_i$  as a surrogate complexity measure of a cofactor representation. The advantage of this approach is that an  $\ell_1$ -minimal solution of a linear system over  $\mathbb{Q}$  can be found efficiently using linear programming. Additionally, we use the effective description of the syzygy module provided by signature-based algorithms to reduce the size of the linear system that we have to consider.

Based on Corollary 3.8, it suffices to consider only cofactor representations up to a degree bound when computing minimal representations. Here, we, more generally,

restrict to representations with signature less than a designated bound  $\sigma \in M(\Sigma)$ . Since the module ordering is assumed to be fair, this ensures that we work in a finite dimensional vector space. If the module ordering is also compatible with the degree, i.e., if  $\deg(\alpha) \leq \deg(\beta)$  implies  $\alpha \leq \beta$ , this includes all cofactor representations of degree  $< \deg(\sigma)$ .

So, formally, we seek a minimal element w.r.t.  $\|\cdot\|_1$  in the set

$$R(f, \sigma) := \{\alpha \in R(f) \mid \mathfrak{s}(\alpha) < \sigma\}$$

of cofactor representations of  $f$  up to signature  $\sigma \in M(\Sigma)$ . We denote the set of all such  $\ell_1$ -minimal elements by  $R_1(f, \sigma)$ . Analogously, we let  $R_0(f, \sigma)$  be the set of all minimal elements w.r.t.  $\|\cdot\|_0$  in  $R(f, \sigma)$ .

The results in this section rely on the fact that we have some  $\alpha \in R(f, \sigma)$ . However, in general, for  $\sigma$  too small, the set  $R(f, \sigma)$  can be empty, even if  $R(f) \neq \emptyset$ . To resolve this issue, we assume that we have a cofactor representation  $\alpha \in R(f)$  and that  $\sigma$  is chosen so that  $\sigma > \mathfrak{s}(\alpha)$ . Note that this assumption, in particular, implies that  $f \in (f_1, \dots, f_r)$ . Such  $\alpha$  can be obtained, for example, by reducing  $f$  to zero using a (partial) labeled Gröbner basis and keeping track of the reductions. With this in mind, we assume that  $R(f, \sigma) \neq \emptyset$ .

In the following, we describe Algorithm 3, which allows to compute an element in the set  $R_1(f, \sigma)$ . To this end, we denote by  $I^{[\Sigma]}$  the labeled module generated by  $f_1, \dots, f_r$  and by  $H_\sigma$  a Gröbner basis of  $\text{Syz}(I^{[\Sigma]})$  up to signature  $\sigma$ .

The general idea of Algorithm 3 is still to reduce the problem of computing sparse cofactor representations to computing certain solutions of a linear system. However, instead of choosing all polynomials  $a f_i b$  to form the linear system like Algorithm 1 does, we use the information provided by  $H_\sigma$  to trim this set. More precisely, we find a finite set of module monomials  $B = \{\mu_1, \dots, \mu_d\} \subseteq M(\Sigma)$  such that  $R_i(f, \sigma)$ ,  $i = 0, 1$ , has non-empty intersection with the  $\mathbb{Q}$ -vector space generated by  $B$  and then only consider the polynomials  $\{\bar{\mu}_1, \dots, \bar{\mu}_d\}$  to form the linear system. Furthermore, we now no longer seek a sparsest solution of the resulting system but an  $\ell_1$ -minimal solution, which can be found with linear programming.

It remains to discuss how to find a suitable basis  $B$  and how to translate the problem of finding  $\ell_1$ -minimal solutions of a linear system into a linear programming problem.

#### 4.1. Finding a suitable basis $B$

Algorithm 1 essentially uses the basis  $B = \{a \varepsilon_i b \mid a, b \in \langle X \rangle, i = 1, \dots, r, \mathfrak{s}(a \varepsilon_i b) < \sigma\}$ , which leads to finite dimensional, yet infeasibly large, linear systems. Using a Gröbner basis of  $\text{Syz}(I^{[\Sigma]})$  up to signature  $\sigma$ , we can drastically reduce the dimension of the search space. To this end, we extend the notion of rewriting to module elements.

**Definition 4.1.** Let  $\alpha, \gamma \in \Sigma$  and  $a, b \in \langle X \rangle$  such that  $\text{supp}(\alpha) \cap \text{supp}(a \gamma b) \neq \emptyset$ . For every  $c \in \mathbb{Q}$ , we say that  $\alpha$  can be rewritten to  $\alpha + c a \gamma b$  by  $\gamma$ .

Furthermore, we say that  $\alpha$  can be rewritten to  $\beta$  by  $H \subseteq \Sigma$  if there are  $\beta_0, \dots, \beta_d \in \Sigma$ ,  $\beta_d = \alpha$ ,  $\beta_0 = \beta$  and  $\gamma_1, \dots, \gamma_d \in H$  such that  $\beta_k$  can be rewritten to  $\beta_{k-1}$  by  $\gamma_k$  for all  $k = 1, \dots, d$ .

With this, we can state a module version of Lemma 3.6. We note that we state all results in this section for both the  $\ell_0$ -“norm” and the  $\ell_1$ -norm to emphasize that they hold for both complexity measures likewise and that the restriction to  $\|\cdot\|_1$  only comes later for the linear programming.

**Lemma 4.2.** *Let  $i \in \{0, 1\}$ . Furthermore, let  $\alpha \in R(f, \sigma)$ ,  $\alpha_i \in R_i(f, \sigma)$ , and let  $H_\sigma$  be a Gröbner basis of  $\text{Syz}(I^{[\Sigma]})$  up to signature  $\sigma$ . Then  $\alpha$  can be rewritten to  $\alpha_i$  by  $H_\sigma$ . In particular, this rewriting can be done so that the signature of every rewriter  $a_j \gamma_j b_j$  is less than  $\sigma$ .*

To prove Lemma 4.2, we make use of the fact that the  $\ell_0$ -“norm” and the  $\ell_1$ -norm are linear for elements of disjoint support.

**Lemma 4.3.** *For  $\alpha, \beta \in \Sigma$  with  $\text{supp}(\alpha) \cap \text{supp}(\beta) = \emptyset$ , we have  $\|\alpha + \beta\|_i = \|\alpha\|_i + \|\beta\|_i$  for  $i = 0, 1$ .*

*Proof of Lemma 4.2.* The difference  $\alpha - \alpha_i$  is a syzygy with signature  $< \sigma$ . Since  $H_\sigma$  is a Gröbner basis of  $\text{Syz}(I^{[\Sigma]})$  up to signature  $\sigma$ , there exist  $d \in \mathbb{N}$  and  $\gamma_j \in H_\sigma$ ,  $c_j \in \mathbb{Q}$ ,  $a_j, b_j \in \langle X \rangle$  such that  $\alpha_i = \alpha - \sum_{j=1}^d c_j a_j \gamma_j b_j$  and  $s(a_j \gamma_j b_j) \leq \max\{s(\alpha), s(\alpha_i)\} < \sigma$  for all  $j$ . Now, we essentially follow the proof of Lemma 3.6 and perform induction on  $d$ .

The case  $d = 0$  is clear. Assume now that  $d > 0$  and that the result is proven for all pairs  $(\alpha, \alpha_i)$  such that  $\alpha - \alpha_i$  has a representation with  $d - 1$  terms. Let  $\beta = \sum_{j=1}^d c_j a_j \gamma_j b_j$ . If  $\beta = 0$ , we are done since  $\alpha = \alpha_i$ . So assume  $\beta \neq 0$ , which implies  $\|\beta\|_i > 0$ . Then we must have  $\text{supp}(\alpha) \cap \text{supp}(\beta) \neq \emptyset$ , as otherwise Lemma 4.3 would yield the contradiction  $\|\alpha\|_i = \|\alpha - \beta\|_i = \|\alpha\|_i + \|\beta\|_i > \|\alpha\|_i \geq \|\alpha_i\|_i$ , where the last inequality follows from the minimality of  $\|\alpha_i\|_i$ . Thus, we have  $\text{supp}(\alpha) \cap \text{supp}(a_j \gamma_j b_j) \neq \emptyset$  for some  $1 \leq j \leq d$ . W.l.o.g. assume  $j = d$ . Hence,  $\alpha$  can be rewritten to  $\beta_{d-1} = \alpha - c_d a_d \gamma_d b_d$  by  $\gamma_d \in H_\sigma$ . Note that  $s(a_d \gamma_d b_d) < \sigma$ . Since  $\beta_{d-1} - \alpha_i = \sum_{j=1}^{d-1} c_j a_j \gamma_j b_j$  has a representation with  $d - 1$  terms, the induction hypothesis implies that  $\beta_{d-1}$  can be rewritten to  $\alpha_i$  by  $H_\sigma$  using only rewriters  $a_j \gamma_j b_j$  with signature  $< \sigma$ .  $\square$

Lemma 4.2 says that any  $\alpha \in R(f, \sigma)$  can be rewritten to each element in  $R_i(f, \sigma)$ ,  $i = 0, 1$ , by  $H_\sigma$  using only rewriters with signature bounded by  $\sigma$ . Consequently, to find a suitable basis  $B$ , it suffices, starting from some  $\alpha$ , to only choose those syzygies that can appear in such rewriting sequences. Finding these elements is a purely combinatorial problem that can be solved without performing any actual rewriting steps. This leads to Algorithm 2, in which we collect precisely all those relevant syzygies. Algorithm 2 can be considered as an adaptation of the symbolic preprocessing in the F4 algorithm [Fau99]. In the following, for  $V \subseteq \Sigma$ , let  $\text{supp}(V) = \bigcup_{\gamma \in V} \text{supp}(\gamma)$ .

**Proposition 4.4.** *Algorithm 2 terminates and is correct.*

*Proof.* The conditions on the elements in new ensure that only module monomials smaller than  $\sigma$  are inserted into `todo`. Furthermore, each monomial is processed at most once. Consequently, termination follows from the fact that there are only finitely many monomials smaller than  $\sigma$  (recall that  $\leq$  is fair). Correctness follows from Lemma 4.2.  $\square$

---

**Algorithm 2:** Finding relevant syzygies

---

**Input:**  $\alpha \in R(f, \sigma)$ ,  $H_\sigma$  a Gröbner basis of  $\text{Syz}(I^{[\Sigma]})$  up to signature  $\sigma$

**Output:**  $V \subseteq \text{Syz}(I^{[\Sigma]})$  such that  $R_i(f, \sigma) \subseteq \alpha + \text{span}_{\mathbb{Q}}(V)$ ,  $i = 0, 1$

```
1  $V \leftarrow \emptyset$ ;  
2  $\text{todo} \leftarrow \text{supp}(\alpha)$ ;  
3  $\text{done} \leftarrow \emptyset$ ;  
4 while  $\text{todo} \neq \emptyset$  :  
5   select  $\mu \in \text{todo}$ , remove it, and add it to  $\text{done}$ ;  
6    $\text{new} \leftarrow \{\alpha\gamma b \mid a, b \in \langle X \rangle, \gamma \in H_\sigma, \mu \in \text{supp}(\alpha\gamma b), s(\alpha\gamma b) < \sigma\}$ ;  
7    $\text{todo} \leftarrow \text{todo} \cup (\text{supp}(\text{new}) \setminus \text{done})$ ;  
8    $V \leftarrow V \cup \text{new}$ ;  
9 return  $V$ ;
```

---

Using Algorithm 2, we can set  $B = \text{supp}(\alpha) \cup \text{supp}(V)$  as a basis of the search space, where  $V$  is the output of the algorithm given  $\alpha$  and  $H_\sigma$  as input. In many cases, this set is small enough to reasonably work with.

#### 4.2. Detecting redundant syzygies

As an optional step, we can remove redundant elements from  $V$  before forming the basis  $B$  in order to obtain a smaller basis, and thus, a smaller linear program to solve. More precisely, since we only want to compute one element in  $R_i(f, \sigma)$ ,  $i = 0, 1$ , we can remove syzygies as long as we can ensure that there remains at least one rewriting sequence from  $\alpha$  to at least one element in  $R_i(f, \sigma)$ . We mention two basic techniques that turned out useful in practice.

The first technique allows to remove syzygies from  $V$  that consist mostly of terms that appear in no other element. Such syzygies cannot lead to simpler representations. To make this statement precise, for  $W \subseteq V$  and  $\beta = \sum_j c_j \mu_j \in W$  with  $c_j \in \mathbb{Q}$ ,  $\mu_j \in M(\Sigma)$ , we denote

$$\begin{aligned}\beta_U &:= \sum_j c_j \mu_j \text{ with } j \text{ such that } \mu_j \notin \text{supp}((V \cup \{\alpha\}) \setminus \{\beta\}), \\ \beta_W &:= \sum_j c_j \mu_j \text{ with } j \text{ such that } \mu_j \in \text{supp}((V \cup \{\alpha\}) \setminus W).\end{aligned}$$

Intuitively, the element  $\beta_U$  contains all those terms of  $\beta$  that are unique to  $\beta$  and that appear in no other element of  $V \cup \{\alpha\}$ , and  $\beta_W$  contains those terms that appear in  $\beta$  as well as in elements outside  $W$ .

**Proposition 4.5.** *Let  $i \in \{0, 1\}$ . Furthermore, let  $\alpha \in R(f, \sigma)$  and  $V \subseteq \text{Syz}(I^{[\Sigma]})$  such that*

$$(\alpha + \text{span}_{\mathbb{Q}}(V)) \cap R_i(f, \sigma) \neq \emptyset.$$

*If  $W \subseteq V$  satisfies  $\|\beta_W\|_i \leq \|\beta_U\|_i$  for all  $\beta \in W$ , then*

$$(\alpha + \text{span}_{\mathbb{Q}}(V \setminus W)) \cap R_i(f, \sigma) \neq \emptyset.$$

Proposition 4.5 provides a sufficient condition for a subset  $W \subseteq V$  to be redundant. In order to prove this, we need the following two lemmas. The first one states that the required property of  $W$  extends to the whole linear span. To this end, we extend the definition of  $\beta_U$  and  $\beta_V$  to elements  $\beta = \sum_j b_j \beta_j \in \text{span}_{\mathbb{Q}}(W)$ , where  $b_j \in \mathbb{Q}$  and  $\beta_j \in W$ , by  $\beta_U := \sum_j b_j \beta_{j,U}$  and  $\beta_V := \sum_j b_j \beta_{j,V}$ . Note that, in general, these definitions depend on the representation of  $\beta$  in terms of the elements in  $W$ ; different linear combinations of the same element can yield different definitions of  $\beta_U$  and  $\beta_V$ . Therefore, to obtain an unambiguous definition, we assume that, for every element  $\beta \in \text{span}_{\mathbb{Q}}(W)$ , one particular representation in terms of  $W$  has been fixed, and this is the representation that we use to compute  $\beta_U$  and  $\beta_V$ . We note that, for all our arguments, the particular choice of the representation does not matter. It is only important that, for each  $\beta$ , the elements  $\beta_U$  and  $\beta_V$  are computed with respect to the same representation.

**Lemma 4.6.** *Let  $V, W$  be as in Proposition 4.5. If  $\beta \in \text{span}_{\mathbb{Q}}(W)$ , then  $\|\beta_V\|_i \leq \|\beta_U\|_i$ .*

*Proof.* Write  $\beta = \sum_j b_j \beta_j$  with nonzero  $b_j \in \mathbb{Q}$  and  $\beta_j \in W$ . By assumption  $\|\beta_{j,V}\|_i \leq \|\beta_{j,U}\|_i$  for all  $j$ . Furthermore, all  $\beta_{j,U}$  have pairwise disjoint supports as they consist of the monomials that are unique to each  $\beta_j$ . So Lemma 4.3 implies that  $\|\cdot\|_i$  is linear on linear combinations of the  $\beta_{j,U}$ . Using this and the triangular inequality, we get with  $c_j = 1$  if  $i = 0$  and  $c_j = |b_j|$  if  $i = 1$ :

$$\begin{aligned} \|\beta_V\|_i &\leq \sum_j \|b_j \beta_{j,V}\|_i = \sum_j c_j \|\beta_{j,V}\|_i \\ &\leq \sum_j c_j \|\beta_{j,U}\|_i = \sum_j \|b_j \beta_{j,U}\|_i = \|\beta_U\|_i. \end{aligned} \quad \square$$

The second lemma provides a lower bound on the norm of sums  $\gamma + \beta \in \alpha + \text{span}_{\mathbb{Q}}(W)$ .

**Lemma 4.7.** *Let  $\alpha, V, W$  be as in Proposition 4.5. If  $\beta \in \text{span}_{\mathbb{Q}}(W)$  and  $\gamma \in \alpha + \text{span}_{\mathbb{Q}}(V \setminus W)$ , then  $\|\gamma + \beta\|_i \geq \|\gamma\|_i - \|\beta_V\|_i + \|\beta_U\|_i$ .*

*Proof.* Let  $\beta' = \beta - (\beta_U + \beta_V)$ . By definition,  $\beta_U$  and  $\beta'$  have pairwise different supports. Furthermore,  $\gamma + \beta_V$  does not share a monomial with  $\beta_U$  and  $\beta'$  as  $\text{supp}(\gamma + \beta_V) \subseteq \text{supp}((V \cup \{\alpha\}) \setminus W)$  and all monomials of  $\beta$  that lie in this set are collected in  $\beta_V$ . Therefore, Lemma 4.3 and the inverse triangle inequality imply

$$\begin{aligned} \|\gamma + \beta\|_i &= \|\gamma + \beta_V\|_i + \|\beta_U\|_i + \|\beta'\|_i \\ &\geq \|\gamma + \beta_V\|_i + \|\beta_U\|_i \geq \|\gamma\|_i - \|\beta_V\|_i + \|\beta_U\|_i. \end{aligned} \quad \square$$

*Proof of Proposition 4.5.* We claim that removing, if present, elements from  $W$  from a representation  $\delta \in \alpha + \text{span}_{\mathbb{Q}}(V)$  cannot increase the norm. This implies the assertion of the proposition. To prove our claim, write  $\delta$  as  $\delta = \gamma + \beta$  with  $\gamma \in \alpha + \text{span}_{\mathbb{Q}}(V \setminus W)$  and  $\beta \in \text{span}_{\mathbb{Q}}(W)$ . Now, Lemma 4.6 and 4.7, show that  $\|\delta\|_i = \|\gamma + \beta\|_i \geq \|\gamma\|_i - \|\beta_V\|_i + \|\beta_U\|_i \geq \|\gamma\|_i$ .  $\square$

The redundancy test provided by Proposition 4.5 is computationally fairly cheap to check for a given set  $W \subseteq V$ . However, finding suitable candidates for  $W$  is not so trivial. In our implementation, we test all singletons  $\{\beta\} \subseteq V$  and all subsets  $\{\beta, \gamma\} \subseteq V$  where  $\text{supp}(\beta) \cap \text{supp}(\gamma) \neq \emptyset$  and where  $\beta$  and  $\gamma$  consist, to at least a third, of unique

monomials that appear in no other element in  $V$ . Empirically, this provided the best trade-off between efficiency in applying the criterion and the effect it had on pruning  $V$ .

The second method does not directly allow to detect redundant elements in  $V$ . Instead it can be considered as an auxiliary technique that can cause additional applications of Proposition 4.5. The idea is to replace elements in  $V$  by linear combinations so that the number of occurrences of certain monomials is reduced. In particular, by exploiting the fact that

$$\text{span}_{\mathbb{Q}}(V \cup \{\alpha - \beta, \gamma + \beta\}) = \text{span}_{\mathbb{Q}}(V \cup \{\alpha - \beta, \alpha + \gamma\}), \quad (2)$$

we can reduce the number of occurrences of  $\beta$  at the cost of increasing the occurrences of  $\alpha$ .

In our implementation, we apply this technique to all binomial syzygies  $\mu - \sigma \in V$ . After removing all occurrences of  $\sigma$ , Proposition 4.5 allows to delete the binomial syzygy from  $V$ . Additionally, we apply (2) randomly to elements  $\alpha - \beta$  where  $\|\beta\|_i > c\|\alpha\|_i$  for fixed  $c > 1$ . Often, this process triggers further invocations of Proposition 4.5 to remove elements from  $V$ . Table 1 shows the efficiency of the two methods presented in this section.

#### 4.3. Translation into linear program

Once we have obtained a reasonable basis of module monomials  $B = \{\mu_1, \dots, \mu_d\}$  such that the  $\mathbb{Q}$ -vector space generated by  $B$  has non-empty intersection with  $R_i(f, \sigma)$ ,  $i = 0, 1$ , we can set up a linear system  $A\mathbf{y} = \mathbf{b}$ , where  $A$  is the matrix of size  $s \times d$ , with  $s = |\bigcup_j \text{supp}(\bar{\mu}_j)|$ , whose  $j$ th column contains the coefficients of  $\bar{\mu}_j$ , associating to each row of  $A$  a monomial  $m \in \bigcup_j \text{supp}(\bar{\mu}_j)$ . Similarly,  $\mathbf{b}$  is a vector of size  $s$  containing the coefficients of  $f$ . The matrix  $A$  bears resemblance to the matrices appearing in Gröbner basis computations such as the F4 algorithm, aside from two main differences. In Gröbner basis computations, polynomials are encoded as the rows of a matrix and the columns have to be ordered w.r.t. the (polynomial) monomial ordering. In our approach, polynomials are encoded as the columns and the order of the rows is irrelevant.

Every solution  $\mathbf{y}$  of  $A\mathbf{y} = \mathbf{b}$  corresponds to a cofactor representation of  $f$  with support in  $B = \{\mu_1, \dots, \mu_d\} \subseteq M(\Sigma)$ . To see this, we denote by  $A_{i,j}$  the entries of  $A$  and by  $\mathbf{b}_i$  and  $\mathbf{y}_j$  the coordinates of  $\mathbf{b}$  and  $\mathbf{y}$  respectively. Furthermore, let  $m_i \in \bigcup_j \text{supp}(\bar{\mu}_j)$  be the monomial that is associated to the  $i$ th row of  $A$  and the  $i$ th coordinate of  $\mathbf{b}$ . Then we can write  $f = \sum_{i=1}^s \mathbf{b}_i m_i$  and  $\bar{\mu}_j = \sum_{i=1}^s A_{i,j} m_i$ , for  $j = 1, \dots, d$ , and we see that

$$f = \sum_{i=1}^s \mathbf{b}_i m_i = \sum_{i=1}^s \left( \sum_{j=1}^d A_{i,j} \mathbf{y}_j \right) m_i = \sum_{j=1}^d \mathbf{y}_j \left( \sum_{i=1}^s A_{i,j} m_i \right) = \sum_{j=1}^d \mathbf{y}_j \bar{\mu}_j,$$

showing that  $\sum_{j=1}^d \mathbf{y}_j \bar{\mu}_j \in R(f, \sigma)$  is a cofactor representation of  $f$ .

Moreover, every  $\ell_i$ -minimal solution of  $A\mathbf{y} = \mathbf{b}$  corresponds to an element in  $R_i(f, \sigma)$ . As noted before, computing  $\ell_0$ -minimal, i.e., sparsest, solutions is NP-hard. Therefore, we restrict ourselves to the case  $i = 1$  and consider the problem

$$(P_1) : \quad \min_{\mathbf{y}} \|\mathbf{y}\|_1 \quad \text{subject to} \quad A\mathbf{y} = \mathbf{b},$$

where  $\|\mathbf{x}\|_1 = \sum_j |\mathbf{x}_j|$ . It is well-known that  $(P_1)$  can be recast as a linear program, see e.g. [CDS01, Sec. 3.1]. A linear program (in standard form) [Sch98] is an optimization problem for  $\mathbf{v} \in \mathbb{Q}^t$  of the form

$$(LP): \quad \min_{\mathbf{v}} \mathbf{c}^T \mathbf{v} \quad \text{subject to} \quad U\mathbf{v} = \mathbf{w}, \quad \mathbf{v} \geq 0,$$

where  $\mathbf{v} \geq 0$  is to be understood component-wise. The problem  $(P_1)$  can be equivalently formulated as a linear program by setting

$$t = 2d, \quad \mathbf{c}^T = (1, \dots, 1), \quad U = (A \mid -A), \quad \mathbf{v} = \begin{pmatrix} \mathbf{p} \\ \mathbf{q} \end{pmatrix}, \quad \mathbf{w} = \mathbf{b},$$

with vectors  $\mathbf{p}, \mathbf{q} \in \mathbb{Q}^d$ . This linear program can then be solved efficiently using the simplex algorithm [Dan51] or interior-point methods [PW00] and a solution  $\mathbf{y}$  of  $(P_1)$  is given by  $\mathbf{y} = \mathbf{p} - \mathbf{q}$ .

#### 4.4. Putting everything together

Finally, we combine the results of the previous sections to form Algorithm 3 for computing an element in  $R_1(f, \sigma)$ . In the algorithm,  $I^{[\Sigma]}$  denotes the labeled module generated by  $f_1, \dots, f_r$ .

---

#### Algorithm 3: $\ell_1$ -minimal cofactor representation

---

**Input:**  $f_1, \dots, f_r \in \mathbb{Q}\langle X \rangle$ ,  $f \in (f_1, \dots, f_r)$ ,  $\sigma \in M(\Sigma)$ ,  $\alpha \in R(f, \sigma)$

**Output:** an element in  $R_1(f, \sigma)$

- 1  $H_\sigma \leftarrow$  Gröbner basis of  $\text{Syz}(I^{[\Sigma]})$  up to signature  $\sigma$ ;
  - 2  $V \leftarrow$  apply Algorithm 2 to  $\alpha$  and  $H_\sigma$ ;
  - 3  $V \leftarrow$  prune  $V$  using the techniques from Section 4.2;
  - 4  $\{\mu_1, \dots, \mu_d\} \leftarrow \text{supp}(V \cup \{\alpha\})$ ;
  - 5  $A \leftarrow$  matrix with columns containing the coefficients of  $\bar{\mu}_1, \dots, \bar{\mu}_d$ ;
  - 6  $\mathbf{b} \leftarrow$  vector containing the coefficients of  $f$ ;
  - 7  $\mathbf{v} \leftarrow$  solution of the linear program  $(LP)$  with
$$\mathbf{c}^T = (1, \dots, 1), \quad U = (A \mid -A), \quad \mathbf{v} = \begin{pmatrix} \mathbf{p} \\ \mathbf{q} \end{pmatrix}, \quad \mathbf{w} = \mathbf{b};$$
  - 8 **return**  $\sum_{i=1}^d (\mathbf{p}_i - \mathbf{q}_i) \mu_i$ ;
- 

**Theorem 4.8.** *Algorithm 3 terminates and is correct.*

*Proof.* Termination follows from the fact that  $H_\sigma$  can be computed in finite time by Theorem 2.6, and from Proposition 4.4. Correctness follows from the discussions in Section 4.1, 4.2 and 4.3.  $\square$

*Remark 4.9.* The complexity of Algorithm 3, in the worst case, is polynomial in the size of the matrix built, or singly exponential in the size of the input. This is the same complexity as one would achieve in Algorithm 1 by using  $\ell_1$ -relaxation and an LP-solver instead of the last step.

In particular, it is not clear how much the use of signatures and the pruning techniques can reduce the size of the matrix in the worst case, so they do not change the worst-case complexity bounds. However, they have a large impact on the complexity in practice, as the experimental results in Section 5 demonstrate.

*Remark 4.10.* Algorithm 3 weighs each monomial  $\mu_i$  equally by a weight of 1. It is also possible to weigh the monomials differently by changing the vector  $\mathbf{c}$  so that  $\mathbf{c}_i$  encodes the weight of  $\mu_i$ . This allows, for example, to weigh monomials by their degree, yielding representations that prefer monomials with small degree. In this case, the output of the algorithm is no longer guaranteed to be in  $R_1(f, \sigma)$ , as witnessed by the example ROL-3 discussed in Section 5.

#### 4.5. Special case: totally unimodular matrices

In general, the output of Algorithm 3 need not be a sparsest representation of  $f$  up to signature  $\sigma$ , i.e., it need not be an element in  $R_0(f, \sigma)$ . In this section, we discuss a special case when this is indeed true. To this end, we consider the linear system  $\mathbf{A}\mathbf{y} = \mathbf{b}$  constructed in Algorithm 3. We are interested in situations where the augmented matrix  $(\mathbf{A} \mid \mathbf{b})$  is *totally unimodular* as defined below.

**Definition 4.11.** A matrix  $T \in \{-1, 0, 1\}^{m \times n}$  is called *totally unimodular* if every square submatrix of  $T$  has determinant 0 or  $\pm 1$ .

**Theorem 4.12.** *Let  $\mathbf{A}$  and  $\mathbf{b}$  be as constructed in Algorithm 3. If the augmented matrix  $(\mathbf{A} \mid \mathbf{b})$  is totally unimodular, then the output of Algorithm 3 is an element in  $R_0(f, \sigma)$ .*

In order to prove the theorem, we take a closer look at the coefficients of the sparsest and  $\ell_1$ -minimal solutions of  $\mathbf{A}\mathbf{y} = \mathbf{b}$ . It is well-known that totally unimodular coefficient matrices and integer right-hand sides yield integer optima for linear programs [Sch98, Cor. 19.1a]. The following lemma, extends this statement under slightly stricter assumptions.

**Lemma 4.13.** *Let the augmented matrix  $(\mathbf{A} \mid \mathbf{b})$  be totally unimodular. If  $\mathbf{A}\mathbf{y} = \mathbf{b}$  is solvable, then any sparsest or  $\ell_1$ -minimal solution  $\mathbf{y}$  satisfies  $\mathbf{y} \in \{-1, 0, 1\}^d$ .*

*Proof.* Since removing linearly dependent rows does not change the solution set of a solvable system, we can assume that  $\mathbf{A}$  has full row rank  $s = \text{rank}(\mathbf{A})$ .

*Sparsest solution.* The columns of  $\mathbf{A}$  corresponding to the nonzero entries of  $\mathbf{y}$  have to be linearly independent (otherwise there would exist a sparser solution). We can extend them by further columns of  $\mathbf{A}$  to obtain an invertible  $s \times s$  matrix  $\mathbf{A}'$ . Then  $\mathbf{A}'\mathbf{y}' = \mathbf{b}$ , where  $\mathbf{y}'$  contains those coordinates of  $\mathbf{y}$  that correspond to the columns of  $\mathbf{A}$  that are in  $\mathbf{A}'$ . By assumption  $\det(\mathbf{A}') = \pm 1$ . Furthermore, the matrix  $\mathbf{A}'_i$  obtained by replacing the  $i$ th column of  $\mathbf{A}'$  by  $\mathbf{b}$  is – up to permutation of columns – a submatrix of  $(\mathbf{A} \mid \mathbf{b})$ . Consequently,  $\det(\mathbf{A}'_i) \in \{-1, 0, 1\}$  and applying Cramer's rule shows  $\mathbf{y}'_i = \frac{\det(\mathbf{A}'_i)}{\det(\mathbf{A}')} \in \{-1, 0, 1\}$ . Then the result follows since any coordinate of  $\mathbf{y}$  which does not appear in  $\mathbf{y}'$  has to be zero.

*$\ell_1$ -minimal solution.* We consider the equivalent linear program ( $LP$ ) and note that  $\mathbf{y} \in \{-1, 0, 1\}^d$  if and only if  $\mathbf{v} \in \{0, 1\}^{2d}$ . If  $\mathbf{v}$  is a solution of ( $LP$ ), then it has to be a basic feasible solution. This means  $\|\mathbf{v}\|_0 \leq s$  and that the columns of  $\mathbf{U}$  that correspond



to the nonzero coordinates of  $\mathbf{v}$  can be extended to an invertible  $s \times s$  submatrix  $U'$  of  $U$ . Since  $(U \mid \mathbf{b}) = (A \mid -A \mid \mathbf{b})$  is totally unimodular, the same arguments as in the other case show that  $\mathbf{v} \in \{-1, 0, 1\}^{2d}$ , and the statement follows from the non-negativity constraint of  $(LP)$ .  $\square$

Using this lemma, we can now prove Theorem 4.12.

*Proof of Theorem 4.12.* By construction, the system  $A\mathbf{y} = \mathbf{b}$  has a solution. For  $i = 0, 1$ , let  $\alpha_i$  be the module element corresponding to an  $\ell_i$ -minimal solution of the system. Note that, again by construction,  $\alpha_i \in R_i(f, \sigma)$ . By Lemma 4.13,  $\alpha_i$  contains only nonzero coefficients  $\pm 1$ , which implies that  $\|\alpha_i\|_0 = \|\alpha_i\|_1$  for  $i = 0, 1$ , and the result follows.  $\square$

In most applications, all polynomials involved are of the form  $a - b$  with  $a, b \in \langle X \rangle \cup \{0\}$  encoding identities of operators of the form  $A = B$ . Such polynomials are called *pure difference binomials*. The following corollary of Theorem 4.12 ensures that Algorithm 3 computes a sparsest representation up to signature  $\sigma$  provided that the input polynomials are pure difference binomials.

**Corollary 4.14.** *Let  $f, f_1, \dots, f_r \in \mathbb{Q}\langle X \rangle$  be pure difference binomials,  $\sigma \in M(\Sigma)$ , and  $\alpha \in R(f, \sigma)$ . Given these elements as input, the output of Algorithm 3 is an element in  $R_0(f, \sigma)$ .*

*Proof.* Let  $A, \mathbf{b}$  as constructed in Algorithm 3. By assumption on  $f, f_1, \dots, f_r$ , each column of  $(A \mid \mathbf{b})$  contains at most one entry  $+1$  and at most one entry  $-1$  with all other entries being 0. Each square submatrix  $U$  of  $(A \mid \mathbf{b})$  either contains a zero column (then  $U$  is singular), a column with one nonzero entry (then expansion of  $\det(U)$  along this column yields inductively  $\det(U) \in \{-1, 0, 1\}$ ), or each column of  $U$  contains exactly one entry  $+1$  and one entry  $-1$  (then  $\mathbf{1}^T U = 0$  showing that  $U$  is singular). Thus,  $(A \mid \mathbf{b})$  is totally unimodular and the result follows from Theorem 4.12.  $\square$

*Example 4.15.* We revisit Example 3.13. All polynomials that appear in this example are pure difference binomials. Hence, Corollary 4.14 implies that Algorithm 3 yields a sparsest cofactor representation up to the used signature bound  $\sigma$ . In particular, if a degree-compatible module ordering is used and  $\sigma$  is chosen so that  $\deg(\sigma) > 7$ , then by Corollary 3.8 the computed representation is a sparsest one (independent of any bound).

Applying Algorithm 3 to Example 3.13, with the cofactor representation given in (1) and a suitable signature bound  $\sigma$ , yields again (1), showing that this is a sparsest cofactor representation. The basis used to form the linear system only consists of 300 elements, compared to the 88 672 that Algorithm 1 would need.

## 5. Experimental results

We have written a prototype implementation of Algorithm 3 for SAGEMATH<sup>1</sup> using our package `signature_gb`<sup>1</sup> for the signature-based computations and the IBM ILOG CPLEX optimization studio [IBM23] for the linear programming.

<sup>1</sup>Available at <https://clemenshofstadler.com/software/>

In Table 1, we compare the weight of cofactor representations computed by Algorithm 3 to those found by other approaches. In particular, we compare our algorithm to tracing standard Gröbner basis computations and reductions, and to tracing reductions done with a signature Gröbner basis.

As benchmark examples, we recover short proofs of the following (recent) results in operator theory on the Moore-Penrose inverse.

- SVD encodes [Hog13, Ch. 5.7 Fact 4], which provides a formula for the Moore-Penrose inverse of a matrix in terms of the matrix's singular value decomposition.
- ROL encodes the implication  $(2) \Rightarrow (1)$  in [KDC07, Thm. 3], which provides a sufficient condition for the identity  $(AB)^\dagger = B^\dagger A^\dagger$  to hold, where  $X^\dagger$  is the Moore-Penrose inverse of an element in a ring with involution.
- ROL- $n$  encodes the implication  $(n) \Rightarrow (1)$  in [DD10, Thm. 2.1]. This family provides several sufficient conditions for the identity  $(AB)^\dagger = B^\dagger A^\dagger$  to hold, where  $X^\dagger$  is the Moore-Penrose inverse of a bounded operator on Hilbert spaces.
- Hartwig- $n$  encodes the implication  $(n) \Rightarrow (1)$  in [CIHHP<sup>+</sup>21, Thm. 2.3]. This family provides several sufficient conditions for the identity  $(ABC)^\dagger = C^\dagger B^\dagger A^\dagger$  to hold, where  $X^\dagger$  is the Moore-Penrose inverse of an element in a ring with involution.
- Ker encodes part of [RP87, Thm. 1], which characterizes the existence of Moore-Penrose inverses in additive categories with involution in terms of kernels of morphisms.
- SMW encodes [Den11, Thm. 2.1], which generalizes the Sherman–Morrison–Woodbury formula in terms of the Moore-Penrose inverse.
- Sum encodes [Li08, Lem. 1], which provides a sufficient condition for the identity  $(A + B)^\dagger = A^\dagger + B^\dagger$  to hold, where  $X^\dagger$  is the Moore-Penrose inverse of an element in a  $C^*$ -algebra.

These operator statements are translated into polynomial ideal membership analogous to how Theorem 3.12 and its proof are translated in Example 3.13. In particular, each identity of operators is translated into a noncommutative polynomial by uniformly replacing each basic operator by a unique noncommutative indeterminate and by forming the difference of the left- and right-hand side of the identity. With this translation, the assumptions of the operator statement form the generators of the ideal and the claimed identity becomes the polynomial whose ideal membership has to be verified. We refer to [RRHP21] and [Hof23, Sec. 5.1] for a more detailed description of the translation and a theoretical justification of this approach.

For all examples,  $\leq_{\text{deglex}}$  is used in combination with the degree-compatible ordering  $\leq_{\text{DoPoT}}$  for the signature-based computations.

The first columns of Table 1 contain information about the ideals that arise when translating the operator statements. In particular, we list the number of generators of each ideal and their maximal degree. Moreover, in the column for Algorithm 3, we provide information on the used signature bound. A value  $n$  in this column indicates

Example	#gens	deg	GB	SigGB	Algo. 3 (bound)	w/o pruning	w/ pruning	ratio $\neq 0$
SVD	32	3	51	39	25 (10)	$127\text{ k} \times 397\text{ k}$	$117\text{ k} \times 326\text{ k}$	0.82
ROL	28	5	80	39	30 (12)	$22\text{ k} \times 102\text{ k}$	$22\text{ k} \times 55\text{ k}$	0.54
ROL-2	28	5	20	21	15 (12)	$24\text{ k} \times 107\text{ k}$	$23\text{ k} \times 59\text{ k}$	0.55
ROL-3	28	5	49	44	31 (12)	$19\text{ k} \times 87\text{ k}$	$18\text{ k} \times 46\text{ k}$	0.52
ROL-4	28	5	59	46	33 (12)	$68\text{ k} \times 236\text{ k}$	$64\text{ k} \times 136\text{ k}$	0.57
ROL-5	28	5	28	30	22 (12)	$33\text{ k} \times 134\text{ k}$	$31\text{ k} \times 79\text{ k}$	0.59
ROL-6	28	5	39	39	30 (12)	$22\text{ k} \times 99\text{ k}$	$21\text{ k} \times 54\text{ k}$	0.54
ROL-7	40	9	85	23	17 (12)	$18\text{ k} \times 86\text{ k}$	$17\text{ k} \times 45\text{ k}$	0.52
ROL-8	44	7	241	19	17 (12)	$258\text{ k} \times 965\text{ k}$	$242\text{ k} \times 548\text{ k}$	0.57
Hartwig-4	23	15	316	54	46 (18)	$353\text{ k} \times 1\,756\text{ k}$	$334\text{ k} \times 1\,340\text{ k}$	0.76
Hartwig-5	26	15	99	43	35 (17)	$407\text{ k} \times 1\,654\text{ k}$	$392\text{ k} \times 1\,305\text{ k}$	0.79
Hartwig-6	24	15	86	33	29 (17)	$218\text{ k} \times 967\text{ k}$	$215\text{ k} \times 771\text{ k}$	0.80
Ker	12	3	49	34	23 (12)	$50\text{ k} \times 142\text{ k}$	$50\text{ k} \times 129\text{ k}$	0.90
SMW	36	7	63	42	39 (12)	$44\text{ k} \times 114\text{ k}$	$42\text{ k} \times 91\text{ k}$	0.80
Sum	20	3	313	178	85 (9)	$11\text{ k} \times 18\text{ k}$	$10\text{ k} \times 16\text{ k}$	0.92

Table 1: Comparison of weights of cofactor representations computed by standard Gröbner bases (GB), by signature Gröbner bases (SigGB), and by Algorithm 3 (Algo. 3). Size comparison of the coefficient matrix  $A$  (rounded to thousands) in Algorithm 3 with and without applying the pruning techniques from Sec. 4.2.

that we consider only cofactor representations of degree  $< n$ . The degree bounds were chosen so that the computation would finish for the larger examples Hartwig- $n$  and ROL-8 within about 90 minutes on a regular laptop and for the remaining smaller examples within a few minutes. We note that these degree bounds are strictly smaller than those that Corollary 3.8 yields, but the latter were computationally infeasible. Nevertheless, Table 1 shows that Algorithm 3 still allows to find sparser representations for all considered examples.

Apart from the last three (Ker, SMW, Sum), all benchmark examples only consist of pure difference binomials. For those, Corollary 4.14 implies that the representations computed by Algorithm 3 are the sparsest up to the respective degree bounds. For the remaining examples, the algorithm can still be used to find  $\ell_1$ -minimal representations, which are heuristically also sparse, but without guarantee that they are the sparsest.

We also tested an adapted version of Algorithm 3 as described in Remark 4.10 that minimizes the total number of symbols appearing in a cofactor representation. For most benchmark examples, the thereby computed representations have the same (minimal) weight as those found with the standard version of the algorithm, but the total number of symbols decreases by up to 15%. Only for ROL-3 does the weight increase by one, while the number of symbols decreases from 196 to 172. We note that this representation is no longer  $\ell_1$ -minimal.

In the last columns of Table 1, we compare the size of the matrix  $A$  constructed in Algorithm 3 with and without applying the pruning techniques discussed in Section 4.2. We also list the ratio between the number of nonzero entries in the pruned matrix and the number of nonzero entries in the original matrix. As the table shows, in some examples the size of the resulting linear system can be reduced drastically, cutting the number of nonzero entries almost in half.

We note that the obtained matrices can still be very large, even after applying the pruning techniques, however they are typically extremely sparse, which allows to work with them reasonably well. In particular, if the input only consists of pure difference binomials, then each column contains at most two nonzero entries. Thus, for example, the largest matrix appearing in our benchmark examples, the one for Hartwig-5, only contains a total of about 2.6 million nonzero entries, corresponding to a density of nonzero entries of roughly 0.0005 %.

## Acknowledgements

C. Hofstadler was supported by the Austrian Science Fund (FWF) grant P 32301. T. Verron was supported by the Austrian Science Fund (FWF) grant P 34872. We thank Georg Regensburger for valuable remarks which helped to improve the presentation of this work a lot. We are also grateful to the anonymous referees for their helpful comments and for pointing us to relevant literature.

## References

- [BHR23] Klara Bernauer, Clemens Hofstadler, and Georg Regensburger. How to Automate Proofs of Operator Statements: Moore-Penrose Inverse; A Case Study. In *Proceedings of CASC 2023*, pages 39–68, 2023.
- [BR22a] Andrei A. Bulatov and Akbar Rafiey. The ideal membership problem and abelian groups. In *Proceedings of STACS 2022*, volume 219 of *LIPICs*, pages 18:1–18:16, 2022.
- [BR22b] Andrei A. Bulatov and Akbar Rafiey. On the Complexity of CSP-Based Ideal Membership Problems. In *Proceedings of STOC 2022*, pages 436–449, 2022.
- [CDS01] Scott Shaobing Chen, David L. Donoho, and Michael A. Saunders. Atomic Decomposition by Basis Pursuit. *SIAM Rev.*, 43:129–159, 2001.
- [CHRR20] Cyrille Chenavier, Clemens Hofstadler, Clemens G. Raab, and Georg Regensburger. Compatible rewriting of noncommutative polynomials for proving operator identities. In *Proceedings of ISSAC 2020*, pages 83–90, 2020.
- [CIHHP<sup>+</sup>21] Dragana S. Cvetković-Ilić, Clemens Hofstadler, Jamal Hossein Poor, Jovana Milošević, Clemens G. Raab, and Georg Regensburger. Algebraic proof methods for identities of matrices and operators: improvements of Hartwig’s triple reverse order law. *Appl. Math. Comput.*, 409, 2021.
- [CT05] Emmanuel J. Candes and Terence Tao. Decoding by Linear Programming. *IEEE Trans. Inform. Theory*, 51:4203–4215, 2005.
- [CW92] Ronald R. Coifman and M. Victor Wickerhauser. Entropy-Based Algorithms for Best Basis Selection. *IEEE Trans. Inform. Theory*, 38:713–718, 1992.

- [Dan51] George B. Dantzig. Maximization of a linear function of variables subject to linear inequalities. *Activ. Anal. Proc. Alloc.*, 13:339–347, 1951.
- [DD10] Dragan S. Djordjević and Nebojša Č. Dinčić. Reverse order law for the Moore-Penrose inverse. *J. Math. Anal. Appl.*, 361:252–261, 2010.
- [Den11] Chun Yuan Deng. A generalization of the Sherman-Morrison-Woodbury formula. *Appl. Math. Lett.*, 24:1561–1564, 2011.
- [dKS13] Carlos d’Andrea, Teresa Krick, and Martin Sombra. Heights of varieties in multiprojective spaces and arithmetic Nullstellensätze. *Ann. Sci. Éc. Norm. Supér.*, 46:549–627, 2013.
- [Don06] David L. Donoho. Compressed Sensing. *IEEE Trans. Inform. Theory*, 52:1289–1306, 2006.
- [EF17] Christian Eder and Jean-Charles Faugère. A survey on signature-based algorithms for computing Gröbner bases. *J. Symbolic Comput.*, 80:719–784, 2017.
- [Fau99] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases (F4). *J. Pure Appl. Algebra*, 139:61–88, 1999.
- [FS12] Jean-Charles Faugère and Jules Svartz. Solving polynomial systems globally invariant under an action of the symmetric group and application to the equilibria of N vortices in the plane. *Proceedings of ISSAC 2012*, 2012.
- [FSEDV16] Jean-Charles Faugère, Mohab Safey El Din, and Thibaut Verron. On the complexity of computing Gröbner bases for weighted homogeneous systems. *J. Symbolic Comput.*, 76:107 – 141, 2016.
- [FSS13] Jean-Charles Faugère, Mohab Safey El Din, and Pierre-Jean Spaenlehauer. On the complexity of the Generalized MinRank Problem. *J. Symbolic Comput.*, 55:30–58, 2013.
- [FSS14] Jean-Charles Faugère, Pierre-Jean Spaenlehauer, and Jules Svartz. Sparse Gröbner Bases: the Unmixed Case. In *Proceedings of ISSAC 2014*, pages 178–185, 2014.
- [GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., 1979.
- [Hof23] Clemens Hofstadler. *Noncommutative Gröbner bases and automated proofs of operator statements*. PhD thesis, Johannes Kepler University Linz, Austria, 2023. Available at <https://resolver.obvsg.at/urn:nbn:at:at-ubl:1-67821>.
- [Hog13] Leslie Hogben. *Handbook of Linear Algebra*. CRC press, 2013.

- [HRR19] Clemens Hofstadler, Clemens G. Raab, and Georg Regensburger. Certifying operator identities via noncommutative Gröbner bases. *ACM Commun. Comput. Algebra*, 53:49–52, 2019.
- [HV22] Clemens Hofstadler and Thibaut Verron. Signature Gröbner bases, bases of syzygies and cofactor reconstruction in the free algebra. *J. Symbolic Comput.*, 113:211–241, 2022.
- [HV23] Clemens Hofstadler and Thibaut Verron. Signature Gröbner Bases in Free Algebras over Rings. In *Proceedings of ISSAC 2023*, pages 298–306, 2023.
- [HW94] J. William Helton and John J. Wavrik. Rules for computer simplification of the formulas in operator model theory and linear systems. In *Nonselfadjoint operators and related topics*, pages 325–354. Springer, 1994.
- [IBM23] IBM ILOG CPLEX Optimization Studio. *Version*, 22.1(1987–2023), 2023.
- [Jel05] Zbigniew Jelonek. On the effective Nullstellensatz. *Invent. Math.*, 162:1–17, 2005.
- [KDC07] Jerry J. Koliha, Dragan Djordjević, and Dragana Cvetković. Moore–Penrose inverse in rings with involution. *Linear Algebra Appl.*, 426:371–381, 2007.
- [Kin19] Michael Kinyon. Proof simplification and automated theorem proving. *Philos. Trans. Roy. Soc. A*, 377, 2019.
- [Kol88] Janos Kollar. Sharp Effective Nullstellensatz. *J. Amer. Math. Soc.*, 1:963–975, 1988.
- [Lai22] Pierre Lairez. Axioms for a theory of signature bases. *arXiv preprints*, arXiv:2210.13788v1, 2022.
- [Li08] Yuan Li. The Moore–Penrose inverses of products and differences of projections in a  $C^*$ -algebra. *Linear Algebra Appl.*, 428:1169–1177, 2008.
- [LSAZ20] Viktor Levandovskyy, Hans Schönemann, and Karim Abou Zeid. LETTERPLACE – a Subsystem of SINGULAR for Computations with Free Algebras via Letterplace Embedding. In *Proceedings of ISSAC 2020*, pages 305–311, 2020.
- [Mas21] Monaldo Mastrolilli. The Complexity of the Ideal Membership Problem for Constrained Problems Over the Boolean Domain. *ACM Trans. Algorithms*, 17:1–29, 2021.
- [May89] Ernst Mayr. Membership in Polynomial Ideals over  $\mathbb{Q}$  Is Exponential Space Complete. In *STACS 89: 6th Annual Symposium on Theoretical Aspects of Computer Science Paderborn*, pages 400–406, 1989.

- [MM82] Ernst W. Mayr and Albert R. Meyer. The Complexity of the Word Problems for Commutative Semigroups and Polynomial Ideals. *Adv. in Math.*, 46:305–329, 1982.
- [Mor16] Teo Mora. *Solving Polynomial Equation Systems IV: Volume 4, Buchberger Theory and Beyond*, volume 158. Cambridge University Press, 2016.
- [MT17] Ernst W. Mayr and Stefan Toman. *Complexity of Membership Problems of Different Types of Polynomial Ideals*, pages 481–493. Springer International Publishing, 2017.
- [MZ93] Stéphane G. Mallat and Zhifeng Zhang. Matching Pursuits With Time-Frequency Dictionaries. *IEEE Trans. on Signal Process.*, 41:3397–3415, 1993.
- [PT16] Tonnian Pitassi and Iddo Tzameret. Algebraic Proof Complexity: Progress, Frontiers and Challenges. *arXiv preprints*, arXiv:1607.00443, 2016.
- [PW00] Florian A. Potra and Stephen J. Wright. Interior-point methods. *J. Comput. Appl. Math.*, 124:281–302, 2000.
- [RP87] Donald W. Robinson and Roland Puystjens. Generalized inverses of morphisms with kernels. *Linear Algebra Appl.*, 96:65–86, 1987.
- [RRHP21] Clemens G. Raab, Georg Regensburger, and Jamal Hossein Poor. Formal proofs of operator identities by a single formal computation. *J. Pure Appl. Algebra*, 225, 2021.
- [Sch98] Alexander Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, 1998.
- [SL20] Leonard Schmitz and Viktor Levandovskyy. Formally Verifying Proofs for Algebraic Identities of Matrices. In *International Conference on Intelligent Computer Mathematics*, pages 222–236. Springer, 2020.
- [Ver01] Robert Veroff. Finding Shortest Proofs: An Application of Linked Inference Rules. *J. Automat. Reason.*, 27:123–139, 2001.
- [Xiu12] Xingqiang Xiu. *Non-Commutative Gröbner Bases and Applications*. PhD thesis, University of Passau, Germany, 2012. Available at <http://www.opus-bayern.de/uni-passau/volltexte/2012/2682/>.