Contributions to polynomial system solving: Recurrences and Gröbner bases

Jérémy Berthomieu

SCIENCES SORBONNE UNIVERSITÉ

Sorbonne Université

Mémoire d'Habilitation à Diriger des Recherches

Mention Informatique

Contributions to polynomial system solving: Recurrences and Gröbner bases

---

Par : Jérémy Berthomieu

Rapporteurs :

M. Bernard Mourrain,	Directeur de recherche,	INRIA & Université Côte d'Azur
M. Cordian Riener,	Professeur,	Universitetet i Tromsø – Norges arktiske universitet
M. Gilles Villard,	Directeur de recherche,	CNRS & École Normale Supérieure de Lyon

Examinateurs :

M. Alin Bostan,	Directeur de recherche,	INRIA
M. Manuel Kauers,	Professeur,	Johannes Kepler Universität Linz
Mme Fatemeh Mohammadi,	Professeure,	Katholieke Universiteit Leuven, Présidente
M. Mohab Safey El Din,	Professeur,	Sorbonne Université

Date de soutenance : 21 septembre 2023

Thanks

This work would not have been possible without the support of many people.

I would like first to thank Mohab Safey El Din, for giving me so much advice which helped me a lot and of course to be in the jury of this habilitation thesis. I also want to thank all my past and current colleagues in the PolSys group with whom it is a pleasure to collaborate and work: Christian Eder, Jean-Charles Faugère, Vincent Neiger, Ludovic Perret and Guénaël Renault.

Many thanks to Bernard Mourrain, Cordian Riener and Gilles Villard for reviewing this manuscript!

I would also like to thank Alin Bostan for accepting to be in this jury and his advice and support together with the other members of this jury: Manuel Kauers and Fatemeh Mohammadi.

I am lucky to be part of a very nice work environment. Thank you to all the other members of the PolSys group in these last few years: Lorenzo Baldi, Olive Chakraborty, Andrew Ferguson, Jorge García Fontán, Sriram Gopalakrishnan, Dimitri Lesnoff, Rafael Mohr, Hadrien Notarantonio, Pierre Pébereau, Rémi Prébet, Georgy Scholten, Thi Xuan Vu, Trung-Hieu Vu. And thank you to the all the former and current members of the PEQUAN team, in particular Stef Graillat for all his support and the fun talking about our children, and also Dominique Bérériat, Pierre Fortin, Fabienne Jézéquel, Thibault Hilaire Christoph Lauter, Théo Mary, Valérie Ménissier-Morain and Marc Mezzarobba.

Thank you to all the administrative staff of LIP6 for their help.

A little bit further, thank you to the members of MATHEXP and in particular Frédéric Chyzak and Pierre Lairez.

More generally, thank you the whole Computer Algebra community, especially the members of the *GT Calcul Formel*.

This works has been partly supported by joint ANR-FWF ANR-19-CE48-0015 ECARP and ANR-22-CE91-0007 EAGLES projects, the ANR grants ANR-18-CE33-0011 Sesame and ANR-19-CE40-0018 De Rerum Natura projects, European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement N. 813211 (POEMA), grant DIM-RFSI 2021-02–C21/1131 of the Paris Île-de-France Region, grant FA8665-20-1-7029 of the EOARD-AFOSR and PGMO grant CAMiSAdo.

On a personal level, I would like to thank my friends, my family in law, Amélie, Benjamin and Nathalie, and my family, Arnaud, Géraldine, Juliette, Joséphine, Chloé, Thibault, Antoine, Baptiste and my parents for their constant support.

Finally, my deepest thank and love for everything to Camille and our two beautiful sons, Tristan and Paul.

Thanks

Contents

Chapter 1

Introduction

This habilitation thesis deals with the research that I have done since I have been with the PolSys group in Sorbonne Université.

On the one hand, polynomial systems arise in a wide range of areas of scientific domains such as biology [39], chemistry [62], quantum mechanics [85], robotics [91], and computing sciences, including coding theory [94], computer vision [72] and cryptography [51] to cite a few. On the other hand, polynomial system solving is NP-hard, even when the ground field is finite [61, Appendix A7.2]. Moreover, the non-linearity of such systems make reliability issues topical, in particular when complete and exhaustive outputs are required, in the context of numerical algorithms.

Likewise, sequences are a classical mathematical object and computing linear recurrence relations of a multi-indexed sequence or determining the nature of this sequence based on these relations is a fundamental problem in coding theory [65, 94], computer algebra [52, 98, 100] and enumerative combinatorics [28, 30, 31].

Whether it be for solving polynomial systems or for computing or *guessing* linear recurrence relations, one aims to obtain nice generators of an ideal that are able to answer the following questions. Is the number of solutions finite in an algebraic closure of the field of coefficients? How many initial terms and linear recurrence relations do one needs to compute any term of the sequence?

These questions are easily answered when we have a *Gröbner basis* of the ideal at hand. This is why, the main focus of my research is Gröbner bases computations with two main goals: solving polynomial systems exactly and determining all the linear recurrence relations of a multi-indexed sequence.

1.1 Gröbner bases: a fundamental object

Buchberger developed the theory of Gröbner bases and designed a first algorithm to compute them in [35]. Since then, many efficient Gröbner basis algorithms were developed.

While lexicographic Gröbner bases are the tool of choice to represent the solution set of a polynomial system, often, they are the hardest Gröbner bases to compute. For n generic polynomials of degree d in n variables, computing the $\prec_{\text{LEX}}$-Gröbner basis of the ideal they spanned is bounded by $C_1 d^{C_2 n^3}$, see [37], whereas computing the $\prec_{\text{DRL}}$-Gröbner basis of the same ideal is bounded by $C_1 d^{C_2 n^2}$, see [74].

As a caveat to that, Gröbner bases change of orders algorithms have been introduced. They take as an input a Gröbner basis $\mathcal{G}_1$ for a monomial order $\prec_1$ and another monomial order $\prec_2$ and they return $\mathcal{G}_2$, a Gröbner basis of $\langle \mathcal{G}_1 \rangle$ for $\prec_2$. This yields the following framework, in the zero-dimensional case, where $f_1, \ldots, f_s$ are the original polynomials that are given as an input to Buchberger's algorithm [35] or to Faugère's F_4 [47] or F_5 [48] algorithms to compute the $\prec_{\text{DRL}}$-Gröbner basis $\mathcal{G}_{\text{DRL}}$. Then, $\mathcal{G}_{\text{DRL}}$ is converted into the $\prec_{\text{LEX}}$-Gröbner basis $\mathcal{G}_{\text{LEX}}$ using the so-called FGLM algorithm [50] or faster variants like the Sparse-FGLM algorithm [52, 53] or more recently [83] and [20]. This framework allows one to compute the $\prec_{\text{LEX}}$-Gröbner basis in $C_1 d^{C_2 n^2}$ operations as well.

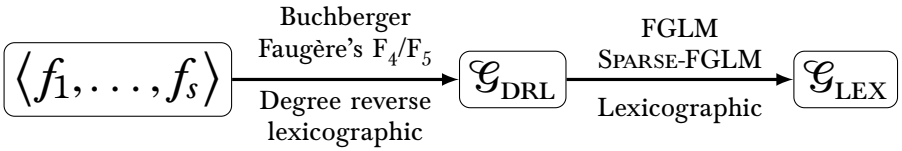

Figure 1.1: Classical Gröbner bases framework for 0-dimensional ideals

Nowadays, many computer algebra software propose implementations of algorithms for computing Gröbner bases, including F_4, FGLM and/or Sparse-FGLM. We can cite Magma [24], Maple [9],

SINGULAR [42], as general software, but also some efficient C library like FGB [49] and MSOLVE [13].

1.2 Solving polynomial systems

Let $\mathbb{K}$ be an algebraic closed field and $n \in \mathbb{N}^*$. We let $\boldsymbol{x} = (x_1, \ldots, x_n)$. We recall that an ideal $I \subseteq \mathbb{K}[\boldsymbol{x}]$ is *zero-dimensional* if the solution set $\left\{\boldsymbol{a} \in \mathbb{K}^n \,\middle|\, \forall\, f \in I,\ f(\boldsymbol{a}) = 0\right\}$ is finite.

With the convention that $x_n \prec_{\text{LEX}} \cdots \prec_{\text{LEX}} x_1$, a $\prec_{\text{LEX}}$-Gröbner basis $\mathcal{G}_{\text{LEX}}$ of I allows one to compute the coordinates of the above solution set. Indeed, for all $1 \leq k \leq n$, $\mathcal{G}_{\text{LEX}} \cap \mathbb{K}[x_k, \ldots, x_n]$ is a $\prec_{\text{LEX}}$-Gröbner basis of $I \cap \mathbb{K}[x_k, \ldots, x_n]$. That is, $\prec_{\text{LEX}}$-Gröbner bases extend the notion of triangular basis for linear systems.

Assuming $\mathcal{G}_{\text{LEX}}$ is reduced, it suffices then to compute the roots of the (unique) polynomial in $\mathcal{G}_{\text{LEX}} \cap \mathbb{K}[x_n]$ to obtain the last coordinate of each solution. Then, to replace x_n by each of this last coordinate in all the polynomials in $\mathcal{G}_{\text{LEX}} \cap \mathbb{K}[x_{n-1}, x_n]$ to find the corresponding second-to-last coordinate of each solution, and so on and so forth.

Furthermore, if I is radical, then after a generic change of coordinates, its reduced $\prec_{\text{LEX}}$-Gröbner basis is said to be in *shape position*, i.e.

$$\mathcal{G}_{\text{LEX}} = \left\{g_n(x_n), x_{n-1} - g_{n-1}(x_n), \ldots, x_1 - g_1(x_n)\right\},$$

with $\deg g_n = \deg I$ and for all $1 \leq k \leq n-1$, $\deg g_k < \deg g_n$. In other words, the last coordinate parameterizes the other coordinates of the solutions.

Thanks to this, computing a $\prec_{\text{LEX}}$-Gröbner basis of a zero-dimensional ideal $\langle f_1, \ldots, f_s \rangle$ is a fundamental step for solving exactly a polynomial system $f_1 = \cdots = f_s = 0$. Using the framework of Figure 1.1, we are led to first compute a $\prec_{\text{DRL}}$-Gröbner basis of this ideal.

Under generic assumptions, the complexity of Gröbner bases change of order from $\prec_{\text{DRL}}$ to $\prec_{\text{LEX}}$ is in $\tilde{O}(D^\omega)$ [83] and $O(tD^2)$ [53], where D is the degree of the ideal and $t \leq D$ is the number of monomials not in x_n in the $\prec_{\text{DRL}}$-monomial basis. This parameter t is well-understood when the ideal is spanned by n generic polynomials of degree d [53, Cor. 5.10].

In many applications, the polynomial system at hands is structured. For instance, in polynomial optimization, critical values of a polynomial map restricted to an algebraic sets are solutions of determinantal systems deriving from maximal minors.

Problem 1 (Complexity of Sparse-FGLM for generic determinantal systems)**:** Is it possible to give asymptotics for t in generic cases for some of these structured systems?

Furthermore, depending on the asymptotics on t and the theoretical or practical values of ω, it is not clear which complexity is the better one between $\tilde{O}(D^\omega)$ [83] and $O(tD^2)$ [53].

Problem 2 (Complexity of Gröbner bases change of order)**:** Under the same generic assumptions, is there an algorithm whose complexity is better than [53, 83] independently on the ratio t/D and on the chosen bound for ω?

In real algebraic geometry, computing sample points in singular real algebraic sets [92] or computing their real dimension [73] require to compute *limits* of critical points of the restriction of a polynomial map to some algebraic set depending on a parameter. This boils down to the computation of saturated ideals. State-of-the-art algorithms for a saturated ideal $I : \langle \varphi \rangle^\infty$ [4, 90] rely on the Gröbner basis computation of an augmented ideal containing I, introducing an extra variable that must be eliminated later. Furthermore, running the F_4 algorithm on this kind of ideal, we can observe that the computations are not regular: there a lot of degree falls.

Problem 3 (Computation of Gröbner bases of saturated ideals)**:** Is it possible to compute a Gröbner basis of $I : \langle \varphi \rangle^\infty$ on the fly from the generators of I without introducing an extra variable?

1.3 Sequences

Let $\boldsymbol{u} \in \mathbb{K}^{\mathbb{N}^n}$ be a n-indexed sequence with values in $\mathbb{K}$, that is $\boldsymbol{u} = (u_{i_1,\ldots,i_n})_{(i_1,\ldots,i_n)\in\mathbb{N}^n}$. There is a natural correspondence between finite linear combinations of terms of $\boldsymbol{u}$ and polynomials in $\mathbb{K}[x_1, \ldots, x_n] = \mathbb{K}[\boldsymbol{x}]$. For $g = \sum_{\boldsymbol{s}\in\mathcal{S}} \gamma_{\boldsymbol{s}} \boldsymbol{x}^{\boldsymbol{s}}$, with $\mathcal{S}$ a finite subset of $\mathbb{N}^n$, we write $\text{Eval}\,(g, \boldsymbol{u}) :=$

$\sum_{s\in\mathcal{S}} \gamma_s u_s$. Since

$$\mathrm{Eval}\left(g\boldsymbol{x}^{\boldsymbol{i}}, \boldsymbol{u}\right) = \sum_{s\in\mathcal{S}} \gamma_s u_{s+i},$$

shifting a linear combination of terms by an index $\boldsymbol{i}$ comes down to multiplying the associated polynomial by $\boldsymbol{x}^{\boldsymbol{i}}$. Observe that Eval is a bilinear operator from $\mathbb{K}[\boldsymbol{x}] \times \mathbb{K}^{\mathbb{N}^n}$ to $\mathbb{K}$.

Definition-Proposition 1.1 (e.g. [12, Def. 2 and Prop. 4])**:** Let $\boldsymbol{u} \in \mathbb{K}^{\mathbb{N}^n}$ be a sequence. A polynomial $g \in \mathbb{K}[\boldsymbol{x}]$ defines a *linear recurrence relation with constant coefficients*, or *C-relation* for short, on $\boldsymbol{u}$ if, and only if, for all $\boldsymbol{i} \in \mathbb{N}^n$, $\mathrm{Eval}\left(g\boldsymbol{x}^{\boldsymbol{i}}, \boldsymbol{u}\right) = 0$.

The set of all such polynomials is an ideal of $\mathbb{K}[\boldsymbol{x}]$ called the *ideal of C-relations of* $\boldsymbol{u}$ and denoted $\mathrm{I_C}(\boldsymbol{u})$.

Definition-Proposition 1.2 (e.g. [11, Def. 2 and Prop. 3])**:** A sequence $\boldsymbol{u} \in \mathbb{K}^{\mathbb{N}^n}$ is said to be *C-finite* if together with a finite number of terms of $\boldsymbol{u}$ and a finite number of C-relations, one can recover all the terms of $\boldsymbol{u}$. This is equivalent to requiring $\mathrm{I_C}(\boldsymbol{u})$ be 0-dimensional.

In some applications, like in combinatorics, error correcting code, but also in computer algebra through polynomial system solving, one must deal with the ideal of C-relations of a n-indexed sequence. In the last two applications, the $\prec_{\mathrm{LEX}}$-Gröbner basis of this ideal is the target and its roots the ultimate goal.

It is unclear, at this stage, if given an ideal $I \in \mathbb{K}[\boldsymbol{x}]$, we can build a sequence $\boldsymbol{u} \in \mathbb{K}^{\mathbb{N}^n}$ such that $\mathrm{I_C}(\boldsymbol{u}) = I$. However, we have this following easy result.

Definition-Proposition 1.3: If $\boldsymbol{u}$ and $\boldsymbol{v}$ in $\mathbb{K}^{\mathbb{N}^n}$ are such that $\mathrm{I_C}(\boldsymbol{u}) \subseteq I$ and $\mathrm{I_C}(\boldsymbol{v}) \subseteq I$, then for any $\lambda \in \mathbb{K}$, $\mathrm{I_C}(\boldsymbol{u} + \lambda\boldsymbol{v}) \subseteq I$, where $\boldsymbol{u} + \lambda\boldsymbol{v} = (u_i + \lambda v_i)_{i\in\mathbb{N}^n}$. Hence, for an ideal I of $\mathbb{K}[\boldsymbol{x}]$, there exists a whole vector subspace of $\mathbb{K}^{\mathbb{N}^n}$, denoted $\mathrm{S_C}(I)$ such that for $\boldsymbol{u} \in \mathrm{S_C}(I)$, $\mathrm{I_C}(\boldsymbol{u}) \subseteq I$.

Remark 1.4: By convention, the *zero sequence* is the only sequence whose ideal of C-relations is $\langle 1\rangle$.

Theorem 1.5: Let $\boldsymbol{u}$ and $\boldsymbol{v}$ in $\mathbb{K}^{\mathbb{N}^n}$. Let $\lambda, \mu \in \mathbb{K}$. Then,

$$\mathrm{I_C}(\boldsymbol{u}) \cap \mathrm{I_C}(\boldsymbol{v}) \subseteq \mathrm{I_C}(\lambda\boldsymbol{u} + \mu\boldsymbol{v}).$$

Proof. Let $g = \sum_{s\in\mathcal{S}} \gamma_s \boldsymbol{x}^{\boldsymbol{s}} \in \mathrm{I_C}(\boldsymbol{u}) \cap \mathrm{I_C}(\boldsymbol{v})$. By linearity, for all $\boldsymbol{i} \in \mathbb{N}^n$, $\mathrm{Eval}\left(g\boldsymbol{x}^{\boldsymbol{i}}, \lambda\boldsymbol{u} + \mu\boldsymbol{v}\right) = \lambda\,\mathrm{Eval}\left(g\boldsymbol{x}^{\boldsymbol{i}}, \boldsymbol{u}\right) + \mu\,\mathrm{Eval}\left(g\boldsymbol{x}^{\boldsymbol{i}}, \boldsymbol{v}\right) = 0$. Hence, $g \in \mathrm{I_C}(\lambda\boldsymbol{u} + \mu\boldsymbol{v})$. □

In this manuscript, most examples will come from sequences such as the following one.

Example 1.6: Let $I = \left\langle (x_2^4 + x_2^3)(x_1^2 + x_2^2 - 1), (x_1 - x_2^2 - x_2)(x_1^2 + x_2^2 - 1)\right\rangle$. It defines the unit cercle with multiplicity 1, through the factor $x_1^2 + x_2^2 - 1$ appearing in both spanning polynomials and two points: the origin $(0,0)$, with multiplicity 3, and $(0,-1)$, with multiplicity 1, as the common vanishing points of $x_2^4 + x_2^3 = x_1 - x_2^2 - x_2 = 0$. Notice that $(0,-1)$ is also on the unit cercle and thus it has multiplicity 2 in total.

Let $\varphi = x_1^4 - x_2^4 - 2x_1^2 + 1 = (x_1^2 + x_2^2 - 1)(x_1^2 - x_2^2 - 1)$ defining the union of the unit circle and the unit hyperbola.

The ideal $J = I : \langle\varphi\rangle^\infty$ defines the Zariski closure of the set difference of the variety defined by I and that defined by φ. The resulting ideal defines the origin, with multiplicity 3, and one can proves that $J = \left\langle x_2^3, x_1 - x_2^2 - x_2\right\rangle$.

Now, let $\boldsymbol{v} \in \mathrm{S_C}(J)$. By the first polynomial, we have $\mathrm{Eval}\left(x_1^i x_2^{j+3}, \boldsymbol{v}\right) = v_{i,j+3} = 0$ for all $(i,j) \in \mathbb{N}^2$. Moreover, by the second polynomial, we have $v_{i+1,j} = \mathrm{Eval}\left(x_1^{i+1}x_2^j, \boldsymbol{v}\right) = \mathrm{Eval}\left(x_1^i x_2^{j+2} + x_1^i x_2^{j+1}, \boldsymbol{v}\right) = v_{i,j+2} + v_{i,j+1}$ for all $(i,j) \in \mathbb{N}^2$. Hence, there exist $\alpha_{0,0}$, $\alpha_{0,1}$ and $\alpha_{0,2}$ such that

$\boldsymbol{v} =$

$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\iddots$
3	0	0	0	0	$\cdots$
2	$\alpha_{0,2}$	0	0	0	$\cdots$
1	$\alpha_{0,1}$	$\alpha_{0,2}$	0	0	$\cdots$
0	$\alpha_{0,0}$	$\alpha_{0,1} + \alpha_{0,2}$	$\alpha_{0,2}$	0	$\cdots$
i_2 \ i_1	0	1	2	3	$\cdots$

Let us recall that not all ideals can be ideals of C-relations. For instance, any sequence $\boldsymbol{u}$, built from $K = \langle x_1^2, x_1x_2, x_2^2\rangle$, satisfies the relation $\mathrm{Eval}\left((u_{0,1}x_1 - u_{1,0}x_2)x_1^i x_2^j, \boldsymbol{u}\right) = u_{0,1}u_{i+1,j} - u_{1,0}u_{i,j+1} = 0$ for all $(i,j) \in \mathbb{N}^2$. We refer to Appendix A for more details. Hence $K \subsetneq \mathrm{I_C}(\boldsymbol{u})$, and K is not an ideal of C-relations. More generally, [32, Prop. 3.3] proves that ideals of C-relations are exactly *Gorenstein ideals* [63, 76] and problems occur only if the ideal has a zero of multiplicity at least 2. The following theorem can also be found in [45, Th. 8.3].

Theorem 1.7: Let $I \subseteq \mathbb{K}[\boldsymbol{x}]$ be a 0-dimensional ideal. The ideal I (resp. ring $\mathscr{R} = \mathbb{K}[\boldsymbol{x}]/I$) is *Gorenstein* if, equivalently,

(1) $\mathscr{R}$ and its dual are isomorphic as $\mathscr{R}$-modules;

(2) there exists a $\mathbb{K}$-linear form τ on $\mathscr{R}$ such that the following bilinear form is non degenerate

$$\begin{aligned}\mathscr{R} \times \mathscr{R} &\to \mathbb{K}\\ (a,b) &\mapsto \tau(a\,b).\end{aligned}$$

In order to be self-contained, we detail in Appendix A the link between an ideal of C-relations I and the general term of a sequence in $\mathrm{S_C}(I)$.

Guessing the ideal of C-relations of a sequence $\boldsymbol{u}$ is the task of computing candidate polynomials in $\mathrm{I_C}(\boldsymbol{u})$ using only finitely many terms of $\boldsymbol{u}$. In particular, we aim at guessing a $\prec$-Gröbner basis of $\mathrm{I_C}(\boldsymbol{u})$ for a given monomial order $\prec$. In the uni-indexed sequence case, guessing the essentially unique C-relation can be modeled through the solving a Hankel system. Yet, the best algorithm, which runs in time quasi-linear in the number of input sequence terms, uses fast polynomial arithmetic and fast extended Euclidean algorithm [34].

In the multi-indexed setting, one can still guess the C-relations using linear algebra techniques on a *multi-Hankel* matrix [11, 12] or a Gram–Schmidt process [81]. The so-called Berlekamp–Massey–Sakata algorithm, due to Sakata [93, 95, 97], guesses the relations using polynomial additions and shifts by a monomial.

Problem 4 (Polynomial arithmetic for guessing C-relations)**:** Can we model the guessing of C-relations of multi-indexed sequences with multivariate polynomial arithmetic, including polynomial multiplications and divisions?

In some applications, for instance the polynomial system solving one through the Sparse-FGLM algorithm, the bottleneck of the algorithm is the computation of the sequence terms, and not the guessing of the C-relations. It is thus essential that only terms strictly needed for the guessing be computed.

Problem 5 (Minimization of number of queries)**:** In the case where queries to the sequence are expensive, can we relate the number of queries to the geometry of the staircase of the output Gröbner basis?

In enumerative combinatorics, like 2D/3D-walks, the sequence comes with a structure: the terms are invariant by the action of a group on the indices, the nonzero terms lie on a cone. In particular, when many sequence terms are, this make the previously mentioned multi-Hankel matrix have a lot of zero rows. This translates into guessed relations that might be fake as we would not guess them with more terms at hand. Thus, we need to build a much larger matrix, for instance with many more rows than columns, to prevent guessing these fake relations.

Problem 6 (Structured guessing)**:** Can we exploit the sparsity of the nonzero terms in order to guess fewer fake relations with fewer queries? What properties do the guess relations have?

1.4 Contributions

1.4.1 Problem 1: Complexity of Sparse-FGLM for generic determinantal systems

In polynomial optimization, critical points methods require to compute the vanishing set of the restriction of a linear map to an algebraic set defined by p generic polynomials by means of the simultaneous vanishing of maximal minors of a truncated Jacobian matrix. More generally, we

define below the class of *generic determinantal sum ideals* that contains this kind of ideals and we provide an asymptotic on the parameter t for them.

Definition 1.8 (see also [10, Def. 8])**:** With $\mathbb{K}$ an infinite field, let $I \subset \mathbb{K}[\boldsymbol{x}]$ be an ideal which is the sum of m polynomials of degree at most d and the maximal minors of a matrix with polynomial entries also of degree at most d. We say that I is a *generic determinantal sum ideal* if the following three conditions hold:

- the ideal I is zero-dimensional and in shape position;
- the Hilbert series of $\mathbb{K}[\boldsymbol{x}]/I$ is

$$H(\tau) = \frac{\det(M(\tau^{d-1}))}{\tau^{(d-1)\binom{m-1}{2}}} \frac{(1-\tau^d)^m (1-\tau^{d-1})^{n-m}}{(1-\tau)^n}$$

  where $M(\tau)$ is the $(m-1)\times(m-1)$-matrix whose (i,j)th entry is $\sum_k \binom{m-i}{k}\binom{n-1-j}{k}\tau^k$;
- for all $e \geq 1$, the Hilbert series of $(\mathbb{K}[\boldsymbol{x}]/I)/\langle x_n^e \rangle$ is equal to the series $(1-\tau)H$ truncated at the first non-positive coefficient.

Theorem 1.9 (see also [10, Th. 2])**:** Let I be a generic determinantal sum ideal of $\mathbb{K}[\boldsymbol{x}]$ as in Definition 1.8. Let $\mathcal{G}_{\text{DRL}}$ be the reduced $\prec_{\text{DRL}}$-Gröbner basis of I. Let t be the number of polynomials in $\mathcal{G}_{\text{DRL}}$ whose leading monomial is divisible by x_n.

Then, for $d = 2$ and $n \geq m$,

$$t = \sum_{k=0}^{m-1} \binom{n-m-1+k}{k} \binom{m}{\lfloor 3m/2 \rfloor - 1 - j}.$$

Moreover, for $d \geq 3$ and $n \to \infty$,

$$t \approx \frac{1}{\sqrt{(n-m)\pi}} \sqrt{\frac{6}{(d-1)^2-1}}\, d^m (d-1)^{n-m} \binom{n-2}{m-1}.$$

1.4.2 Problem 2: Complexity of Gröbner bases change of order

State-of-the-art FGLM algorithms uses multiplication matrices to compute the polynomial in the reduced Gröbner basis for the target monomial order. These are matrices of size the degree of the ideal and with scalar entries. Using a change of paradigm and relying on polynomial matrix arithmetic, we design a new change of order algorithm whose complexity is in $\tilde{O}(t^{\omega-1}D)$ under the same generic assumption as [53]. As a consequence, it is always asymptotically faster than both [53] and [83]. Furthermore, for ω close to 2, the complexity becomes quasi-linear in the size of the input.

Theorem 1.10 (see also [20, Th. 1.1])**:** Let I be a zero-dimensional ideal $\mathbb{K}[\boldsymbol{x}]$ of degree D. Let $\mathcal{G}_{\text{DRL}}$ (resp. $\mathcal{G}_{\text{LEX}}$) be the reduced $\prec_{\text{DRL}}$- (resp. $\prec_{\text{LEX}}$-)Gröbner basis of I and S_{DRL} be the $\prec_{\text{DRL}}$-monomial basis of $\mathbb{K}[\boldsymbol{x}]/I$. Assume that $x_1, \ldots, x_{n-1}$ are in S_{DRL}, and that for all monomials $\mu \in S_{\text{DRL}}$, either $x_n\mu$ is in S_{DRL} or it is the $\prec_{\text{DRL}}$-leading monomial of an element in $\mathcal{G}_{\text{DRL}}$. Assume that I is in shape position. Then, one can compute $\mathcal{G}_{\text{LEX}}$ using $\tilde{O}\left(t^{\omega-1}D\right)$ operations in $\mathbb{K}$, where t is the number of elements of $\mathcal{G}_{\text{DRL}}$ whose $\prec_{\text{DRL}}$-leading monomial is divisible by x_n.

1.4.3 Problem 3: Computation of Gröbner bases of saturated ideals

Given generators $f_1, \ldots, f_s$ of an ideal I and a polynomial φ, we propose an algorithm extending Faugère's F_4 algorithm [47] that computes a partial $\prec_{\text{DRL}}$-Gröbner basis of I and uses this partial information to find polynomials in $I : \langle\varphi\rangle$ not in I using linear algebra. Then, adding these new generators, it resumes the partial $\prec_{\text{DRL}}$-Gröbner basis computation of $I : \langle\varphi\rangle$. Repeating this process, it finds polynomials in $(I : \langle\varphi\rangle) : \langle\varphi\rangle$ and so on and so forth until it stabilizes and has computed the $\prec_{\text{DRL}}$-Gröbner basis of $I : \langle\varphi\rangle^\infty$. This is the F_4SAT algorithm, Algorithm 3.2.

Furthermore, we modify the Sparse-FGLM algorithm so that it takes as an input the reduced $\prec_{\text{DRL}}$-Gröbner basis of an ideal I and perform linear algebra operations in the set of multiples of φ in $\mathbb{K}[\boldsymbol{x}]/I$ in order to compute the reduced $\prec_{\text{LEX}}$-Gröbner basis of the zero-dimensional ideal $I : \langle\varphi\rangle$. This allows us to perform in one step a colon ideal computation and a change of order. Moreover,

this new Algorithm 3.3, Sparse-FGLM-colon, does not assume that I is zero-dimensional, which finds application in polynomial optimization.

Theorem 1.11 (see also [15, Th. 4.12]): Let I be a positive-dimensional ideal of $\mathbb{K}[\boldsymbol{x}]$, let $\mathcal{G}_{\mathrm{DRL}}$ be its reduced $\prec_{\mathrm{DRL}}$-Gröbner basis and S_{DRL} be the associated staircase. Let $\varphi \in \mathbb{K}[\boldsymbol{x}] \setminus I$ such that $I : \langle\varphi\rangle$ is zero-dimensional of degree D' and in shape position.

Let Σ be a finite staircase of size N containing $\operatorname{supp} \operatorname{NF}\left(x_n^i\varphi, \mathcal{G}_{\mathrm{DRL}}, \prec_{\mathrm{DRL}}\right)$ for all $i \in \mathbb{N}$, where $\operatorname{NF}\left(f, G, \prec\right)$ is the normal form of f w.r.t. G and $\prec$.

Let t be the number of monomials σ in Σ such that $x_n\sigma \in \operatorname{LM}_{\prec_{\mathrm{DRL}}}(\mathcal{G}_{\mathrm{DRL}})$ and let u be the number of monomials σ in Σ such that $x_n\sigma \in \left\langle \operatorname{LM}_{\prec_{\mathrm{DRL}}}(I)\right\rangle \setminus \operatorname{LM}_{\prec_{\mathrm{DRL}}}(\mathcal{G}_{\mathrm{DRL}})$.

Then, for a generic choice of vector $\boldsymbol{r} \in \mathbb{K}^N$, Algorithm 3.3 terminates and returns the reduced $\prec_{\mathrm{LEX}}$-Gröbner basis of $I : \langle\varphi\rangle$. To do so, it requires at most $u + n$ normal form computations w.r.t. $\mathcal{G}_{\mathrm{DRL}}$ and $\prec_{\mathrm{DRL}}$ plus $O\left((t + u + n)ND'\right)$ operations in $\mathbb{K}$.

As a byproduct, I designed and implemented in msolve an efficient probabilistic test for the computation of the parametrizations of $x_1, \ldots, x_{n-1}$ w.r.t. x_n in $\sqrt{I}$. It relies on the algorithm of [29, 66] using x_n as the linear form. Since their algorithm assume that the linear form is generic, we cannot assert that x_n parameterizes the other variables. If it does not, the computed parametrizations are meaningless. Hence the necessity for an efficient validity check of the output. Furthermore, the algorithm of [29, 66] builds a sequence before guessing the parametrizations and this sequence building is the actual bottleneck. Thus, the main advantage of this validity check is that it avoids building a brand new sequence. The following lemma gives the main idea.

Lemma 1.12 (see also [15, Lem. 4.15]): Let I be a zero-dimensional ideal of $\mathbb{K}[\boldsymbol{x}]$. Let $\lambda \in \bar{\mathbb{K}}$ be generic. Then, for $1 \leq k \leq n$, $I = I : \langle x_k + \lambda\rangle$.

It suffices then to compute the parametrizations of such $I : \langle x_k + \lambda\rangle$ using the same sequence $\boldsymbol{u}$ as for I. Indeed, $I : \langle x_k + \lambda\rangle$ is the ideal of C-relations of $\boldsymbol{v} = \left(\operatorname{Eval}\left((x_k + \lambda)\boldsymbol{x}^{\boldsymbol{i}}, \boldsymbol{u}\right)\right)_{\boldsymbol{i}\in\mathbb{N}^n}$.

1.4.4 Problem 4: Polynomial arithmetic for guessing C-relations

While the Berlekamp–Massey–Sakata algorithm [93, 95, 97] performs multivariate polynomials additions and multiplications by a monomial to compute a Gröbner basis of the ideals of C-relations, it fails to generalize the extended Euclidean algorithm. Using the mirror of the truncated generating series of the input multi-index sequence, we design an algorithm performing polynomial divisions and normal forms to compute potential C-relations. Furthermore, we exhibit a criterion based on leading monomials to ensure which of these potential C-relations are correct or not. Finally, when called on a uni-index sequence, the algorithm comes down to performing a truncated extended Euclidean algorithm. All in all, we have the following result.

Theorem 1.13 (see also [19, Th. 1]): Let $\boldsymbol{u}$ be a sequence, $\prec$ be a weighted degree monomial order and M be a monomial. Let us assume that the reduced $\prec$-Gröbner basis $\mathcal{G}$ of $\mathrm{I_C}(\boldsymbol{u})$ and its associated staircase S satisfy $\max(S \cup \operatorname{LM}_{\prec}(\mathcal{G})) \preceq M$ and for all $m \preceq M$, $s = \max_{\sigma\preceq M}\{\sigma \mid \sigma m \preceq M\}$, we have $\max(S) \preceq s$. Then, the variant of the Berlekamp–Massey–Sakata algorithm using polynomial arithmetic terminates and computes $\mathcal{G}$ in $O\left(\#S(\#S + \#\mathcal{G})\#\mathcal{T}_{\preceq M}\right)$ operations in the base field, where $\mathcal{T}_{\preceq M}$ is the set of monomials less or equal to M.

1.4.5 Problem 5: Minimization of number of queries

Assuming the first terms of the sequence are generic enough so that no fake C-relations are wrongly guessed, we design an adaptive algorithm that follows the shape of the staircase of the output Gröbner basis. It follows a strategy similar to the FGLM algorithm, which allows us to minimize the number of queries to the sequence.

Theorem 1.14 (see also [19, Th. 26]): Let $\boldsymbol{u}$ be a sequence whose ideal of C-relations is zero-dimensional. Let $\prec$ be a monomial order, $\mathcal{G}$ be the reduced $\prec$-Gröbner basis of $\mathrm{I_C}(\boldsymbol{u})$ and S be the associated staircase.

Assuming the Adaptive Scalar-FGLM algorithm called on $\boldsymbol{u}$ returns a Gröbner basis $\mathcal{G}'$ with staircase S' and $\#S' = \#S$, then $S' = S$ and $\mathcal{G}' = \mathcal{G}$. Furthermore, the algorithm does not need more that $\#2(S \cup \operatorname{LM}_{\prec}(\mathcal{G}))$ sequence queries and $O((\#S + \#\mathcal{G})^2\#2S)$ operations to recover $\mathcal{G}$, where $2S$ is the Minkowski sum of S with itself.

1.4.6 Problem 6: Structured guessing

P-relations, linear recurrence relations with *polynomial*, in the indices, coefficients can be represented by quasi-commutative polynomials. In this setting, we extend the notion of sparse Gröbner basis of [7, 56]. This corresponds to P-relations between terms of the sequence that are all lying in a predefined cone. This finds application for instance in enumerative combinatorics where the indices of the nonzero terms are far from random but are actually all in a given cone. On the one hand, this allows us to reduce the number of sequence queries to guess the relations and on the other hand, this allows us to guess more relations that are correct and fewer that are fake. For instance, for a subsequence of the Gessel walk, using 3 491 sequence terms in the full orthant, we can guess 142 relations amongst which 136 are fake and only 6 are correct. Whereas, taking only sequence terms in a cone allows us to consider sequence terms much further, which in turn allow us to guess more relations. Indeed, with 3 010 terms in a cone of the same sequence, we guess 21 relations and all of them are correct. We refer to Table 5.1 for more details. Let us also notice that these fake relations may hide correct ones as their leading monomials could divide the leading monomials of correct relations.

1.4.7 MSOLVE

MSOLVE [13, 14] is a new efficient open-source C library, developed by Christian Eder[1], Mohab Safey El Din[2] and myself, for solving polynomial systems exactly. It provides implementations of state-of-the-art algorithms for computing Gröbner bases such as F_4 [47] for $\prec_{\text{DRL}}$-Gröbner basis and SPARSE-FGLM [52, 53] for $\prec_{\text{LEX}}$-Gröbner basis of zero-dimensional ideals.

I implemented in MSOLVE the parametrization of the radical [29, 66], its probabilistic test, see Lemma 1.12 and the SPARSE-FGLM-COLON algorithm for computing the $\prec_{\text{LEX}}$-Gröbner basis of $I : \langle\varphi\rangle$ whether I is zero- or positive-dimensional.

1.5 Organization

First, I start by presenting MSOLVE in Chapter 2. I detail the state-of-the-art algorithms that are available in MSOLVE and compare it with other available implementations of Gröbner bases algorithms. This is a joint development with Christian Eder[1] and Mohab Safey El Din[2].

Then, Chapter 3 is dedicated to the design of new algorithms for computing Gröbner bases in order to solve polynomial systems and their complexity analyses. Some of these algorithms are already available in MSOLVE. This is based on joint collaborations with Alin Bostan[3], Christian Eder[1], Andrew Ferguson[4], Vincent Neiger[2] and Mohab Safey El Din[2].

In Chapter 4, I give an overview on different algorithms for guessing Gröbner bases of ideal of C-relations of sequences and their efficiency. This is based on joint work with Brice Boyer[5], Jean-Charles Faugère[3,6] and Mohab Safey El Din[2].

In Chapter 5, I present extensions of works for guessing ideals of relations, but in the case of relations with *polynomial* coefficients. This is again related to polynomial ideals but in a ring where variables *quasi-commute*. This is based on joint work with Jean-Charles Faugère[3,6] and Mohab Safey El Din[2].

Finally, in Chapter 6, I propose my research project for the forthcoming years based on increasing the functionalities of MSOLVE by implementing existing state-of-the-art algorithms and also on designing algorithms for computing Gröbner bases in the commutative and quasi-commutative towards polynomial system solving and applications to sequences.

[1]Rheinland-Pfälzische Technische Universität Kaiserslautern-Landau
[2]Sorbonne Université
[3]INRIA
[4]former Ph.D. student at Sorbonne Université
[5]former Post-doc at Sorbonne Université
[6]CryptoNext Security

MSOLVE [13, 14] is a an open source C library developed by Christian Eder[1], Mohab Safey El Din[2] and myself, for solving multivariate polynomial systems, with a focus on those which have dimension at most 0. It relies on computer algebra methods yielding algebraic parametrizations of their solutions. This allows us to bypass the commonly encountered issues related to accuracy and exhaustively met by numerical methods because of the non-linearity of the input.

This chapter is based on joint work with Christian Eder[1] and Mohab Safey El Din[2] and in particular [13].

2.1 State-of-the-art algorithms

For positive-dimensional ideals, MSOLVE computes their $\prec_{\text{DRL}}$-Gröbner basis using the F_4 algorithm [47]. For 0-dimensional ideals, it returns a parametrization of their solutions. In generic coordinates, and for a radical ideal, this parametrization is close to the $\prec_{\text{LEX}}$-Gröbner basis of the ideal. To compute it, it follows the framework of Figure 1.1. The change of order from $\prec_{\text{DRL}}$ to $\prec_{\text{LEX}}$ is performed thanks to the SPARSE-FGLM algorithm [52, 53]. If the ideal is not radical, but its radical is in shape position, then it computes the parametrizations thanks to [29, 66]. In this latter case, we cannot ensure that the ideal is in generic coordinates, hence we designed and implemented into MSOLVE an efficient probabilistic test to ensure that the computed parametrizations are parametrizations of the radical ideal. The presentation of this test is postponed to Section 3.3, as it relies on algorithms presented in Chapter 3.

Since MSOLVE is also now our tool of choice to implement and validate the new algorithms that we design, some algorithms presented in the following chapters, for instance, F_4SAT, see Section 3.2.2, and SPARSE-FGLM-COLON, see Section 3.2.3, have been implemented and are also available in MSOLVE.

2.1.1 The F_4 algorithm

In [35], Buchberger's algorithm introduced the concept of *critical pairs* for computing Gröbner bases. For two polynomials f_1 and f_2 in a set of generators of an ideal, the critical pair (f_1, f_2) leads to a normal form computation of the *S-polynomial*

$$\text{sp}_{\prec}(f_1, f_2) = \frac{\text{LCM}(\text{LM}_{\prec}(f_1), \text{LM}_{\prec}(f_2))}{\text{LT}_{\prec}(f_1)} f_1 - \frac{\text{LCM}(\text{LM}_{\prec}(f_1), \text{LM}_{\prec}(f_2))}{\text{LT}_{\prec}(f_2)} f_2$$

w.r.t. the current intermediate basis. The *degree* of such a critical pair is defined as $\deg \text{LCM}(\text{LM}_{\prec}(f_1), \text{LM}_{\prec}(f_2))$. Notice that this bounds from above $\deg \text{sp}_{\prec}(f_1, f_2)$.

In Algorithm 2.1 we state the pseudocode of Faugère's F_4 algorithm, highlighting (in red) the main differences to Buchberger's algorithm.

Observe that the termination of the F_4 algorithm only relies on Buchberger's first criterion: $\mathcal{G} = \{g_1, \ldots, g_t\}$ is a $\prec$-Gröbner basis of an ideal I if for all $1 \leq i, j \leq t$, $\text{NF}(\text{sp}_{\prec}(g_i, g_j), \mathcal{G}, \prec) = 0$, see [41, Chap. 2, Sec. 6, Th. 6].

We detail the differences to Buchberger's algorithm.

(1) One can choose several critical pairs at a time, stored in a subset $L \subseteq P$. The so-called *degree strategy* chooses L to be the set of *all* critical pairs of minimal degree for a total degree monomial order, typically $\prec_{\text{DRL}}$.

(2) For all terms of all the generators of the S-polynomials, one searches in the current intermediate Gröbner basis $\mathcal{G}$ for possible reducers. One adds those to L and again search for all of their terms for reducers in $\mathcal{G}$. This is the SymbolicPreprocessing function.

(3) Once all reduction data is collected from the last step, one generates a Macaulay-like matrix

[1] Rheinland-Pfälzische Technische Universität Kaiserslautern-Landau
[2] Sorbonne Université

Input: A list of polynomials $f_1, \ldots, f_s$ spanning an ideal $I \subseteq \mathbb{K}[x]$ and a total degree monomial order $\prec$.
Output: A $\prec$-Gröbner basis $\mathcal{G}$ of I.

$\mathcal{G} := \{f_1, \ldots, f_s\}$.
$P := \{(f_i, f_j) \mid 1 \leq i < j \leq s\}$.
While $P \neq \emptyset$ **do**
    Choose a subset L of P.
    $P := P \setminus L$.
    $L := \text{SymbolicPreprocessing}(L, \mathcal{G})$.
    $L := \text{LinearAlgebra}(L)$.
    For $h \in L$ *with* $\text{LM}_\prec(h) \notin \langle \text{LM}_\prec(\mathcal{G}) \rangle$ **do**
        $P := P \cup \{(g, h) \mid g \in \mathcal{G}\}$.
        $\mathcal{G} := \mathcal{G} \cup \{h\}$.
Return $\mathcal{G}$.

Algorithm 2.1: Faugère's F_4

with columns corresponding to the monomials appearing in L (sorted by $\prec$) and rows corresponding to the coefficients of each polynomial in L. Gaussian Elimination is then applied to reduce now all chosen S-polynomials at once. This is the LinearAlgebra function.

(4) Finally, one adds those polynomials associated to rows in the updated matrix to $\mathcal{G}$ whose leading monomials are not already in $\text{LM}_\prec(\mathcal{G})$.

In order to optimize the algorithm one can now apply Buchberger's product and chain criteria, see [36, 70]. Thus many useless critical pairs are removed before even being added to P and fewer zero rows are computed during the linear algebra part of F_4. Still, in general, there are many zero reductions left.

Different selection strategies yield different behavior of the algorithm. The degree strategy allows one to compute *truncated Gröbner bases* of ideals in case of early terminations.

Definition 2.1: Let $f_1, \ldots, f_s$ be polynomials in $\mathbb{K}[x]$ and $\prec$ be a monomial order. Let μ be a monomial and F_μ be the $\mathbb{K}$-vector subspace of $\langle f_1, \ldots, f_s \rangle$ defined as

$$F_\mu = \left\{ \sum_{i=1}^{s} h_i f_i \;\middle|\; \forall\, 1 \leq i \leq s, \text{LT}_\prec(h_i f_i) \preceq \mu \right\}.$$

Then, $\mathcal{G} \subset F_\mu$ is a μ-truncated $\prec$-Gröbner basis of $\langle f_1, \ldots, f_s \rangle$ if for all $p \in F_\mu$, there exists $g \in \mathcal{G}$ such that $\text{LM}_\prec(g) \mid \text{LM}_\prec(p)$ and $p - \frac{\text{LT}_\prec(p)}{\text{LT}_\prec(g)} g$ is in F_μ.

Observe that taking a triangular basis of F_μ ordered increasingly w.r.t. $\prec$ naturally yields a μ-truncated $\prec$-Gröbner basis thereof.

Proposition 2.2: Let $f_1, \ldots, f_s$ be polynomials in $\mathbb{K}[x]$ and $\prec$ be a monomial order. Let μ be a monomial and F_μ be the $\mathbb{K}$-vector subspace of $\langle f_1, \ldots, f_s \rangle$

$$F_\mu = \left\{ \sum_{i=1}^{s} h_i f_i \;\middle|\; \forall\, 1 \leq i \leq s, \text{LM}_\prec(h_i f_i) \preceq \mu \right\}.$$

A subset $\mathcal{G} = \{g_1, \ldots, g_t\} \subset F_\mu$ is a μ-truncated $\prec$-Gröbner basis of $\langle f_1, \ldots, f_s \rangle$ if, and only if,

$$F_\mu \subseteq G_\mu = \left\{ \sum_{j=1}^{t} h_j g_j \;\middle|\; \forall\, 1 \leq j \leq t, \text{LM}_\prec(h_j g_j) \preceq \mu \right\}.$$

and for all $(g_i, g_j) \in \mathcal{G}^2$ with $i \neq j$, if $\text{LCM}(\text{LM}_\prec(g_i), \text{LM}_\prec(g_j)) \preceq \mu$, then

$$\text{NF}\left(\text{sp}_\prec\left(g_i, g_j\right), \mathcal{G}, \prec\right) = 0.$$

Remark 2.3: *(1)* If $\prec$ is a total degree monomial order, then for $d \in \mathbb{N}$, we can also define a *d-truncated $\prec$-Gröbner basis* as a μ-truncated $\prec$-Gröbner basis for μ the largest monomial of

degree d.

(2) If $\mathcal{G} = \{g_1, \ldots, g_t\}$ is a μ-truncated $\prec$-Gröbner basis of $\langle f_1, \ldots, f_s \rangle$ and

$$\mu \geq \max_{1 \leq i < j \leq t} \operatorname{LCM}(\operatorname{LM}_{\prec}(g_i), \operatorname{LM}_{\prec}(g_j)),$$

then $\mathcal{G}$ is a $\prec$-Gröbner basis of $\langle f_1, \ldots, f_s \rangle$. Indeed, it spans the ideal and by Proposition 2.2, all the S-polynomials reduce to 0 w.r.t. $\mathcal{G}$ and $\prec$. Hence, by Buchberger's first criterion [41, Chap. 2, Sec. 6, Th. 6], it is a $\prec$-Gröbner basis of $\langle f_1, \ldots, f_s \rangle$.

(3) Definition 2.1 depends greatly on the set of generators of the ideal. Consider $f_1 = x^n$, $f_2 = (y-1)^n$ and $f_3 = xy - y - 1$ for $n \in \mathbb{N} \setminus \{0, 1\}$. By Proposition 2.2, $\mathcal{G} = \{f_1, f_2, f_3\}$ is a n-truncated $\prec_{\text{DRL}}$-Gröbner basis of $\langle f_1, f_2, f_3 \rangle$. Yet, this ideal is $\langle 1 \rangle$ hence $\{1\}$ is a m-truncated $\prec_{\text{DRL}}$-Gröbner basis of $\langle 1 \rangle$ for all $m \in \mathbb{N}$.

Lemma 2.4: Let $f_1, \ldots, f_s \in \mathbb{K}[\boldsymbol{x}]$ be the input polynomials of the F_4 algorithm. Let $d \in \mathbb{N}$. Assume that the F_4 algorithm uses the degree selection strategy and that, on line 4, L consists in all the critical pairs of degree d.

If no new polynomial is added to $\mathcal{G}$ on line 10, then $\mathcal{G}$ is a d-truncated $\prec_{\text{DRL}}$-Gröbner basis of $\langle f_1, \ldots, f_s \rangle$.

2.1.2 The Sparse-FGLM algorithm

In this subsection, the input Gröbner basis, $\mathcal{G}_{\text{DRL}}$, is the reduced $\prec_{\text{DRL}}$-Gröbner basis of a zero-dimensional ideal I of degree D. The output is the reduced $\prec_{\text{LEX}}$-Gröbner basis, $\mathcal{G}_{\text{LEX}}$, of I. In [52] and [53, Algo. 3], using [80], the authors observe that the map

$$\begin{aligned} \mathbb{K}[\boldsymbol{x}]/I &\to \mathbb{K}[\boldsymbol{x}]/I \\ f &\mapsto \operatorname{NF}\left(x_n f, \mathcal{G}_{\text{DRL}}, \prec_{\text{DRL}}\right) \end{aligned}$$

given in the basis S_{DRL}, the staircase associated to $\mathcal{G}_{\text{DRL}}$, is represented by a matrix, M_{x_n}, with a special structure given in the following two lemmas.

Lemma 2.5: Let I be a zero-dimensional ideal of $\mathbb{K}[\boldsymbol{x}]$ of degree D, $\mathcal{G}_{\text{DRL}}$ be its reduced $\prec_{\text{DRL}}$-Gröbner basis and $S_{\text{DRL}} = \{\sigma_0, \ldots, \sigma_{D-1}\}$ be its associated staircase. Let M_{x_n} be the matrix of the linear map $f \in \mathbb{K}[\boldsymbol{x}]/I \mapsto \operatorname{NF}\left(x_n f, \mathcal{G}_{\text{DRL}}, \prec_{\text{DRL}}\right) \in \mathbb{K}[\boldsymbol{x}]/I$.

Then, one can build the matrix $M_{x_n} = (m_{i,j})_{0 \leq i,j < D}$ with the following procedure:

- if $x_n \sigma_j = \sigma_k$, then $m_{k,j} = 1$ and for all $0 \leq i < D$, $i \neq k$, $m_{i,j} = 0$;
- otherwise for all $0 \leq i < D$, $m_{i,j}$ is the coefficient of σ_i in $\operatorname{NF}\left(x_n \sigma_j, \mathcal{G}_{\text{DRL}}, \prec_{\text{DRL}}\right)$.

Lemma 2.6 ([52, 53, 80]): Let $f_1, \ldots, f_n$ be generic polynomials of $\mathbb{K}[\boldsymbol{x}]$ of degrees at most d. Let $\mathcal{G}_{\text{DRL}}$ be the reduced $\prec_{\text{DRL}}$-Gröbner basis of $\langle f_1, \ldots, f_n \rangle$. Then, the latter case of Lemma 2.5 only happens if there exists $g \in \mathcal{G}_{\text{DRL}}$ such that $\operatorname{LM}_{\prec_{\text{DRL}}}(g) = x_n \sigma_j$. As a consequence, one has $\operatorname{NF}\left(x_n \sigma_j, \mathcal{G}_{\text{DRL}}, \prec_{\text{DRL}}\right) = x_n \sigma_j - g$.

Following, we can use Wiedemann algorithm [106] on M_{x_n} to recover its minimal polynomial. Furthermore, whenever the reduced $\prec_{\text{LEX}}$-Gröbner basis $\mathcal{G}_{\text{LEX}}$ is in *shape position*, i.e. there exist $g_n, g_{n-1}, \ldots, g_1 \in \mathbb{K}[x_n]$ such that

$$\mathcal{G}_{\text{LEX}} = \left\{ g_n(x_n), x_{n-1} - g_{n-1}(x_n), \ldots, x_1 - g_1(x_n) \right\},$$

and for all $1 \leq k \leq n-1$, $\deg g_k < \deg g_n$, then $g_1, \ldots, g_{n-1}$ can be computed by solving Hankel systems of size D. This can be done using the following two algorithms, Algorithms 2.2 and 2.3.

Proposition 2.7: Let $M \in \mathbb{K}^{D \times D}$ be a matrix with s nonzero coefficients, $\boldsymbol{r} \in \mathbb{K}^D$ be a row-vector and $\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_{n-1} \in \mathbb{K}^D$ be n column-vectors. Then, Algorithm 2.2 is correct and computes the sequences $(\boldsymbol{r} M^i \mathbf{c}_0)_{0 \leq i < 2D}$ and $(\boldsymbol{r} M^i \mathbf{c}_k)_{0 \leq i < D}$ for $1 \leq k \leq n-1$ in $O(sD + nD^2)$ operations in $\mathbb{K}$.

Furthermore, if the vectors $\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_{n-1}$ are vectors of the canonical basis, then this complexity drops to $O((s+n)D)$.

Proposition 2.8: Let $(v_i^{(0)})_{0 \leq i < 2D-1}, (v_i^{(1)})_{0 \leq i < D}, \ldots, (v_i^{(n-1)})_{0 \leq i < D}$ be the first terms of n sequences. Assume that $v^{(0)}$ is linear recurrent of order D. Then, Algorithm 2.3 is correct and computes, for

Input: A matrix $M \in \mathbb{K}^{D\times D}$, a row-vector $\boldsymbol{r} \in \mathbb{K}^D$ and n column-vectors $\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_{n-1} \in \mathbb{K}^D$
Output: $(\boldsymbol{r}M\mathbf{c}_0)_{0\le i<2D}, (\boldsymbol{r}M\mathbf{c}_1)_{0\le i<D}, \ldots, (\boldsymbol{r}M\mathbf{c}_{n-1})_{0\le i<D}$, with $\mathbf{1} = (1,0,\ldots,0)^{\mathrm{T}}$.

$v_0^{(0)} \coloneqq \boldsymbol{r}\mathbf{c}_0, v_0^{(1)} \coloneqq \boldsymbol{r}\mathbf{c}_1, \ldots, v_0^{(n-1)} \coloneqq \boldsymbol{r}\mathbf{c}_{n-1}$.
For i **from** 1 **to** $D-1$ **do**
  $\boldsymbol{r} \coloneqq \boldsymbol{r}M$.
  $v_i^{(0)} \coloneqq \boldsymbol{r}\mathbf{c}_0, v_i^{(1)} \coloneqq \boldsymbol{r}\mathbf{c}_1, \ldots, v_i^{(n-1)} \coloneqq \boldsymbol{r}\mathbf{c}_{n-1}$.
For i **from** D **to** $2D-1$ **do**
  $\boldsymbol{r} \coloneqq \boldsymbol{r}M$.
  $v_i^{(0)} \coloneqq \boldsymbol{r}\mathbf{c}_0$
Return $(v_i^{(0)})_{0\le i<2D}, (v_i^{(1)})_{0\le i<D}, \ldots, (v_i^{(n-1)})_{0\le i<D}$

Algorithm 2.2: Sequences for SPARSE-FGLM

Input: Sequences $(v_i^{(0)})_{0\le i<2D-1}$ and $(v_i^{(k)})_{0\le i<D}$ for $1 \le k \le n-1$ with coefficients in $\mathbb{K}$.
Output: $\gamma_{0,k}, \ldots, \gamma_{D-1,k}$ for $1 \le k \le n-1$ such that for all $0 \le i < D$,

$$v_i^{(k)} = \gamma_{D-1,k} v_{D-1+i}^{(0)} + \cdots + \gamma_{0,k} v_i^{(0)}.$$

For k **from** 1 **to** $n-1$ **do**
  Solve the Hankel linear system

$$\begin{pmatrix} v_0^{(0)} & v_1^{(1)} & \cdots & v_{D-1}^{(0)} \\ v_1^{(0)} & v_2^{(1)} & \cdots & v_D^{(0)} \\ \vdots & \vdots & & \vdots \\ v_{D-1}^{(0)} & v_D^{(1)} & \cdots & v_{2D-2}^{(0)} \end{pmatrix} \begin{pmatrix} \gamma_{0,k} \\ \gamma_{1,k} \\ \vdots \\ \gamma_{D-1,k} \end{pmatrix} = \begin{pmatrix} v_0^{(k)} \\ v_1^{(k)} \\ \vdots \\ v_{D-1}^{(k)} \end{pmatrix}.$$

Return $\gamma_{i,k}$ for $0 \le i < D$ and $1 \le k \le n-1$.

Algorithm 2.3: Hankel system solving for SPARSE-FGLM

all $1 \leq k \leq n-1$, $\gamma_{0,k}, \ldots, \gamma_{D-1,k}$ such that

$$\forall\, 0 \leq i < D,\ v_i^{(k)} = \gamma_{D-1,k} v_{D-1+i}^{(0)} + \cdots + \gamma_{0,k} v_i^{(0)}$$

in $O(\mathsf{M}(D)(n + \log D))$ operations, where $\mathsf{M}(D)$ denote a cost function for multiplying univariate polynomials of degree D with coefficients in $\mathbb{K}$.

We are now in a position to present the SPARSE-FGLM algorithm in the shape position case.

Input: The reduced $\prec_{\text{DRL}}$-Gröbner basis $\mathcal{G}_{\text{DRL}}$ of a zero-dimensional ideal I and its associated staircase S_{DRL} of size D.
Output: The reduced $\prec_{\text{LEX}}$-Gröbner basis of I, if it is in shape position.

1 Build the matrix M as in Lemma 2.5.
2 Pick $\boldsymbol{r} \in \mathbb{K}^D$ a row-vector at random.
3 $\mathbf{1} \coloneqq (1, 0, \ldots, 0)^{\mathrm{T}}$. // the column-vector of coefficients of NF $(1, \mathcal{G}_{\text{DRL}}, \prec_{\text{DRL}})$
4 **For** k **from** 1 **to** $n-1$ **do**
5 Build $\mathbf{c}_k$ the column-vector of coefficients of NF $(x_k, \mathcal{G}_{\text{DRL}}, \prec_{\text{DRL}})$.
6 Compute $(v_i^{(0)})_{0\leq i<2D}, (v_i^{(1)})_{0\leq i<D}, \ldots, (v_i^{(n-1)})_{0\leq i<D})$ with Algorithm 2.2 called on $M, \boldsymbol{r}, \mathbf{1}, \mathbf{c}_1, \ldots, \mathbf{c}_{n-1}$.
7 $g_n \coloneqq$ Berlekamp–Massey$(v_0^{(0)}, \ldots, v_{2D-1}^{(0)})$.
8 **If** $\deg g_n < D$ **then Return** "Not in shape position or bad vector".
9 Compute $g_1 \coloneqq \gamma_{D-1,1} x_n^{D-1} + \cdots + \gamma_{0,1}, \ldots, g_{n-1} \coloneqq \gamma_{D-1,n-1} x_n^{D-1} + \cdots + \gamma_{0,n-1}$ with Algorithm 2.3 called on $(v_i^{(0)})_{0\leq i<2D-1}, (v_i^{(1)})_{0\leq i<D}, \ldots, (v_i^{(n-1)})_{0\leq i<D})$.
10 **Return** $\{g_n(x_n), x_{n-1} - g_{n-1}(x_n), \ldots, x_1 - g_1(x_n)\}$.

Algorithm 2.4: SPARSE-FGLM

Theorem 2.9: Let I be a zero-dimensional ideal of $\mathbb{K}[\boldsymbol{x}]$ of degree D, $\mathcal{G}_{\text{DRL}}$ be its reduced $\prec_{\text{DRL}}$-Gröbner basis and S_{DRL} be its associated staircase. Let M_{x_n} be the matrix of the map $f \in \mathbb{K}[\boldsymbol{x}]/I \mapsto \text{NF}\left(x_n f, \mathcal{G}_{\text{DRL}}, \prec_{\text{DRL}}\right) \in \mathbb{K}[\boldsymbol{x}]/I$ in the monomial basis S_{DRL}.

Let us assume that there are t monomials σ in S_{DRL} such that $x_n\sigma \in \text{LT}_{\prec_{\text{DRL}}}(I)$ and that $x_1, \ldots, x_{n-1} \in S_{\text{DRL}}$, that M_{x_n} is known and that the reduced $\prec_{\text{LEX}}$-Gröbner basis $\mathcal{G}_{\text{LEX}}$ of I is in shape position. Then, Algorithm 2.4 computes $\mathcal{G}_{\text{LEX}}$ in $O(tD^2 + n\mathsf{M}(D))$ operations, where $\mathsf{M}(D)$ denote a cost function for multiplying univariate polynomials of degree D with coefficients in $\mathbb{K}$.

Note that the Berlekamp–Massey algorithm and its faster variants return a factor of g_n, so if the computed polynomial has degree D, i.e. it is the characteristic polynomial of M_{x_n}, then it is also its minimal polynomial. Furthermore, based on a deterministic variant of Wiedemann's algorithm, one can also provide a deterministic variant of this algorithm to recover g_n [53, Algo. 4].

Remark 2.10: In [29, 66], the authors consider the case where an ideal J is not in shape position but its radical $\sqrt{J}$ is. Let us recall that $\sqrt{J} = \left\{f \in \mathbb{K}[\boldsymbol{x}] \,\middle|\, \exists k \in \mathbb{N}, f^k \in J\right\}$, see [41, Chap. 4, Sec. 2, Def. 4]. In that case, the $\prec_{\text{LEX}}$-Gröbner basis of $\sqrt{J}$ can be computed in a similar fashion, it suffices to replace the call to Algorithm 2.3 on line 9 by a call to [66, Algo. 2].

Example 2.11: Let $\mathcal{H}_{\text{DRL}} = \left\{x_2^2 - x_1 + x_2, x_1x_2 - x_1 + x_2, x_1^2 - x_1 + x_2\right\}$ be the $\prec_{\text{DRL}}$-Gröbner basis of the ideal J. Its associated staircase is $T_{\text{DRL}} = \{1, x_2, x_1\}$ and the matrix of the multiplication by x_2 is

$$\begin{array}{c} \begin{matrix} 1 & x_2 & x_1 \end{matrix} \\ \begin{pmatrix} 0 & 0 & 0 \\ 1 & -1 & -1 \\ 0 & 1 & 1 \end{pmatrix} \begin{matrix} 1 \\ x_2 \\ x_1 \end{matrix} \end{array}$$

We build a sequence $\boldsymbol{u}$ with random initial coefficients, e.g. $u_{0,0} = 17$, $u_{1,0} = 5$ and $u_{0,1} = -3$, thanks

to $\mathcal{G}_{\text{DRL}}$:

$$\boldsymbol{u} = \begin{array}{|c|c|c|c|c|c|}\hline \vdots & \vdots & \vdots & \vdots & \vdots & \iddots \\ \hline 3 & 0 & 0 & 0 & 0 & \cdots \\ \hline 2 & -8 & 0 & 0 & 0 & \cdots \\ \hline 1 & 5 & -8 & 0 & 0 & \cdots \\ \hline 0 & 17 & -3 & -8 & 0 & \cdots \\ \hline i_2 \diagdown i_1 & 0 & 1 & 2 & 3 & \cdots \\ \hline \end{array}$$

The subsequence $(u_{0,i_2})_{i_2\in\mathbb{N}}$ satisfies the relation $u_{0,i_2+3} = 0 = \text{Eval}\left(x_2^{i_2+3}, \boldsymbol{u}\right)$ for all $i_2 \in \mathbb{N}$ and no relation of smaller order. Hence $x_2^3 \in J$.

We now solve the Hankel system

$$\begin{pmatrix} u_{0,0} & u_{0,1} & u_{0,2} \\ u_{0,1} & u_{0,2} & u_{0,3} \\ u_{0,2} & u_{0,3} & u_{0,4} \end{pmatrix}\begin{pmatrix}\gamma_0\\ \gamma_1\\ \gamma_2\end{pmatrix} = \begin{pmatrix} u_{1,0} \\ u_{1,1} \\ u_{1,2}\end{pmatrix}$$

$$\begin{pmatrix} 17 & 5 & -8 \\ 5 & -8 & 0 \\ -8 & 0 & 0 \end{pmatrix}\begin{pmatrix}\gamma_0\\ \gamma_1\\ \gamma_2\end{pmatrix} = \begin{pmatrix} -3 \\ -8 \\ 0\end{pmatrix}$$

to find $\gamma_0 = 0$ and $\gamma_1 = \gamma_2 = -1$. Hence, $x_1 - x_2^2 - x_2$ is in J. Observe that we already knew this though. Finally, $\mathcal{H}_{\text{LEX}} = \left\{x_2^3, x_1 - x_2^2 - y\right\}$.

2.2 Experimental results

In order to solve over rational numbers, a multi-modular approach with efficient algorithms is implemented. The F_4 part of the computation is performed with a tracer modulo a first prime p_1. This first computation allows us to learn all the polynomials and their multiples that are needed at each round and also those that are reduced to 0 or that are only used to reduce other polynomials to 0. Modulo subsequent primes, we apply the tracer so that the matrices we handle are optimal: we do not compute any reduction to 0. Furthermore, we know that this multi-modular approach correctly computes the sought $\prec_{\text{DRL}}$-Gröbner basis over $\mathbb{Q}$ as long as its projection modulo p_1 was correctly computed.

We compare MSOLVE with two other computer algebra systems:

- `magma` -V2.23-6 [24]: using the command `Variety()`.
- `maple` -V2019 [9]: using the command `PolynomialSystem()` from the module `SolveTools` with option `engine=groebner`.

All compared implementations use Faugère's F_4 algorithm and variants of the FGLM algorithm and then solve univariate polynomials.

All chosen systems are zero-dimensional with rational coefficients. All computations are done sequentially. First, Table 2.1 compares memory usage of MSOLVE against `maple` and `magma`. It illustrates the low memory usage of MSOLVE. However, we emphasize that MSOLVE is a specialized library while `maple` and `magma` are general purpose computer algebra systems.

Overall, MSOLVE performs very efficiently on a wide range of input systems, using way less memory than its competitors, allowing its users to solve polynomial systems which are not tractable by `maple` and `magma`.

Table 2.2 states various, partly well-known benchmarks, which differ in their specific hardness, like reduction process, pair handling, sparsity of multiplication matrices, etc. It also deals with critical points computations, `CP(d,nv,np)` describes critical points for a system of `np` polynomials in `nv` variables of degree `d`.

For each system we give its degree and if it is radical (all but one are radical). For MSOLVE we give specific timing information, also on the single modular computations: We apply MSOLVE with the `tracer` option, giving also the timings for the first modular computation learning and generating the tracer (`F4 (learn)`) and the timings for the further modular computations applying only the

Examples	MSOLVE	maple	magma	Examples	MSOLVE	maple	magma
Katsura-9	15	271	71	Henrion-5	11	26	23
Katsura-10	25	276	223	Henrion-6	47	171	–
Katsura-11	66	2,279	–	Henrion-7	3,428	–	–
Katsura-12	229	1,123	–	Noon-7	209	419	–
Katsura-13	1,037	–	–	Noon-8	881	1,227	–
Eco-10	27	210	213	CP(3,5,2)	17	525	–
Eco-11	82	428	354	CP(3,6,2)	55	7,885	–
Eco-12	117	1,027	–	CP(3,7,2)	312	–	–
Eco-13	318	8,654	–	CP(4,4,3)	24	635	–
Eco-14	15,748	–	–	CP(4,5,3)	2,065	–	–
Phuoc-1	176	–	–				

Table 2.1: Maximal memory usage given in MB

Examples	System data		MSOLVE single modular computation				MSOLVE overall			maple single modular		Others overall	
	degree	radical	F_4 (prob.)	F_4 (learn)	F_4 (apply)	FGLM	# primes	trace	independent	F_4	FGLM	maple	magma
Katsura-9	256	yes	0.06	0.17	0.03	0.03	83	4.89	7.49	0.10	0.04	104	2,522
Katsura-10	512	yes	0.24	0.81	0.09	0.11	188	43.7	70.5	0.36	0.15	1,278	82,540
Katsura-11	1,024	yes	1.34	6.26	0.45	0.49	388	424	814	1.82	0.74	7,812	–
Katsura-12	2,048	yes	8.61	56.1	3.10	3.96	835	6,262	11,215	8.50	5.40	120,804	–
Katsura-13	4,096	yes	52.8	425	18.9	30.6	1,772	89,390	148,372	60.9	35.7	–	–
Katsura-14	8,192	yes	318	3,336	128	210	3,847	1,308,602	2,007,170	393	271	–	–
Eco-10	256	yes	0.10	0.28	0.05	0.02	161	12.5	21.2	0.14	0.03	26.3	6,520
Eco-11	512	yes	0.39	1.21	0.17	0.07	327	90.3	161	0.56	0.12	312	214,770
Eco-12	1,024	yes	2.25	11,619	1.07	0.34	530	877	1,619	2.97	0.85	4,287	–
Eco-13	2,048	yes	11.7	67.3	6.61	2.12	1,225	12,137	19,553	15.1	6.70	66,115	–
Eco-14	4,096	yes	67.1	516	34.8	25.9	2,670	167,798	254,389	104.8	69.1	–	–
Henrion-5	100	yes	0.01	0.01	0.004	0.01	83	0.71	0.83	0.01	0.01	2.7	93
Henrion-6	720	yes	0.11	0.22	0.07	0.11	612	138	157	0.17	0.16	1,470	–
Henrion-7	5,040	yes	9.55	27.5	6.51	20.46	4,243	117,803	127,456	12.8	27.1	–	–
Noon-7	2,173	yes	1.66	5.3	0.93	1.95	1,305	4,039	5,045	1.97	3.13	432	–
Noon-8	6,545	yes	26.6	153	17.5	72.3	6,462	598,647	640,177	32.4	76.2	5,997	–
Phuoc-1	1,102	no	4.01	4.65	3.42	2.59	753	4,467	5,056	4.60	5.91	–	–
CP(3,5,2)	288	yes	0.03	0.04	0.01	0.03	326	18.1	19.2	0.06	0.05	249	–
CP(3,6,2)	720	yes	0.22	0.59	0.12	0.16	1,042	390	450	0.31	0.22	23,440	–
CP(3,7,2)	1,728	yes	1.97	8.18	1.23	1.20	3,037	9,643	11,511	2.78	2.54	–	–
CP(3,8,2)	4,032	yes	18.5	111.5	12.2	19.6	8,211	269,766	323,838	24.6	25.3	–	–
CP(4,4,3)	576	yes	0.04	0.86	0.03	0.07	339	40.9	41.8	0.08	0.11	916	–
CP(4,5,3)	3,456	yes	3.24	8.60	2.23	4.83	2,747	21,528	23,559	4.33	9.21	–	–
CP(3,6,6)	729	yes	0.18	0.42	0.11	0.15	779	255	294	0.30	0.23	–	–
CP(4,6,6)	4,096	yes	7.70	25.6	5.44	14.09	3,476	71,472	77,941	10.2	16.71	–	–
CP(3,7,7)	2,187	yes	2.49	8.97	1.58	1.86	2,795	12,412	14,375	3.27	3.75	–	–

Table 2.2: Benchmark timings given in seconds.

tracer (`F4 (apply)`). We also use MSOLVE with `independent` modular computations, applying the probabilistic linear algebra in each modular F_4 (`F4 (prob.)`) In any case, we apply the same FGLM implementation. Furthermore, we state the number of primes needed by MSOLVE to solve over $\mathbb{Q}$. For `maple` and `magma` we just give the overall timings. Symbol '–' means that the computation was stopped after waiting more than 10 times the runtime of MSOLVE. For all systems, the bottleneck has been the computation of either a $\prec_{\text{LEX}}$-Gröbner basis in shape position or a rational parametrization of the solution set.

First thing to note is that `magma` is in all instances slower than MSOLVE or `maple`. Although, for some examples, `magma`'s modular F_4 computation is even a bit faster than the other two, `magma`'s bottleneck is both a not optimized FGLM combined with the fact that `magma` seems to lift a $\prec_{\text{LEX}}$-Gröbner basis instead of a rational parametrization (the latter one having in general coefficients of significantly smaller bit size).

For nearly all systems, MSOLVE is faster, sometimes by an order of magnitude, than `maple`. We report on modular timings of `maple` for F_4 (which is based on a probabilistic linear algebra) and FGLM. It appears that MSOLVE's modular implementations of both F_4 and FGLM are faster than the ones in `maple` (with a speed-up sometimes close to 2, sometimes less). It seems that in the multi-modular process, `maple` uses its probabilistic variant of F_4 while MSOLVE takes advantage of its tracer. Also, `maple`'s documentation indicates that on some examples, an algorithm computing a so-called rational univariate representation (preserving multiplicities), instead of FGLM, may be used. It is likely that on most examples we tried, FGLM is not used. Note also that `maple` lifts a whole $\prec_{\text{DRL}}$-Gröbner basis over $\mathbb{Q}$ while MSOLVE avoids this step. Furthermore, our tracer shares some similarities with [86].

There are, of course, few examples, where MSOLVE is not competitive. In particular, for some systems, MSOLVE may need to introduce a generic linear form as previously explained while a rational parametrization can be obtained without it (but up to computing normal forms). Also some systems admit a triangular representation, and/or can be split. It seems that `maple` can detect and sometimes take advantage of such situations. This is typically the case for the `Noon-n` examples.

Chapter 3

Computing Gröbner bases

In this chapter, I present some works that I have done on algorithms for computing Gröbner bases.

First, Section 3.1 is dedicated to change of order algorithms, whether my coauthors and I studied its complexity in some special cases or we designed new algorithms. In Section 3.2, we shall look at the problems of saturating or quotienting ideals. From a geometric point of view, saturating comes down to removing a subvariety and then taking the Zariski closure of the remaining set. Then, in Section 3.3, I present an efficient verification of the computation of the parametrization of the radical of an ideal when we cannot ensure that this radical is in shape position.

This chapter is based on joint work with Alin Bostan[1], Christian Eder[2], Andrew Ferguson[3], Vincent Neiger[4] and Mohab Safey El Din[4], [10, 13, 15, 20].

3.1 Gröbner bases change of order algorithms

Change of order algorithms can be divided into two categories, depending on the dimension of the ideal I. In positive dimension, we can mention the Gröbner walk [38] whose goal is to update step by step the Gröbner basis when "walking" along a path from $\prec_1$ to $\prec_2$ in the Gröbner fan of the ideal. In dimension zero, most algorithms rely on linear algebra in the finite-dimensional quotient algebra, following the idea of [50] yielding the so-called FGLM algorithm.

This original algorithm had a complexity $O(nD^3)$, where D is the degree of the ideal, i.e. the dimension of the algebra over the base field. Let us recall that if the ideal is spanned by n generic polynomials of degree d, then $D = d^n$. Under the assumption that the change of order is from $\prec_{\text{DRL}}$ to $\prec_{\text{LEX}}$ in shape position and some other genericity assumptions, the complexity of the change of order has been improved in [46] to $\tilde{O}(D^\omega)$, where ω is the matrix multiplication exponent. More recently, Neiger and Schost [83] proposed a general change of order algorithm with complexity $\tilde{O}(D^\omega)$ that only requires a *stability assumption* on the $\prec_1$-Gröbner basis and no assumption on the $\prec_2$-Gröbner basis. This stability assumption is for all monomial μ,

$$\forall 1 \leq i < j \leq n, \quad x_j\mu \in \text{LM}_{\prec_1}(I) \Longrightarrow x_i\mu \in \text{LM}_{\prec_1}(I).$$

3.1.1 Complexity of Sparse-FGLM

Together, Propositions 2.7 and 2.8 and Theorem 2.9 yield the complexity of the Sparse-FGLM algorithm in the shape position situation. While it is not easy to estimate exactly the number s of nonzero coefficients of the matrix M_{x_n}, we can give an upper bound of it. Indeed, from the procedure of Lemma 2.5, $s \in O(tD)$, where t is the number of normal forms to perform. Using now Lemma 2.6, we obtain that for an ideal spanned by n generic polynomials of given degrees, t is exactly the number of polynomials in $\mathcal{G}_{\text{DRL}}$ whose leading monomials are divisible by x_n. Therefore, one obtains a bound of the complexity of the Sparse-FGLM algorithm based on the input Gröbner basis.

This opens a new branch of research to estimate t in order to have a finer complexity estimate of the Sparse-FGLM algorithm in different situations. In their paper [53, Cor. 5.11], the authors bound t for generic systems of n polynomials of degree d in $\mathbb{K}[\boldsymbol{x}]$ and find that as d tends to $+\infty$, t grows as $\sqrt{\frac{6}{n\pi}d^{2n-1}}$. This is to compare with D that grows as d^n. More recently, new estimates for t have been given, especially for polynomial systems that have some of their generators as minors of a multivariate polynomial matrix.

[1] INRIA
[2] Rheinland-Pfälzische Technische Universität Kaiserslautern-Landau
[3] former Ph.D. student at Sorbonne Université
[4] Sorbonne Université

In a joint work with Alin Bostan[5], Andrew Ferguson[6] and Mohab Safey El Din[7], we study a case of polynomial ideals that encompass systems defining the critical points of the restriction of a linear map to an algebraic set defined by p generic polynomials by means of the simultaneous vanishing of maximal minors of a truncated Jacobian matrix. For instance, we consider $f_1, \dots, f_m \in \mathbb{K}[\boldsymbol{x}]$, the Jacobian matrix

$$\mathcal{J} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \vdots & & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{pmatrix}$$

of φ_1, the projection on the x_1-axis, $f_1, \dots, f_m$ and the ideal $I = \langle f_1, \dots, f_m \rangle + \langle \text{MaxMinors } \mathcal{J} \rangle$.

Definition 1.8 (see also [10, Def. 8])**:** With $\mathbb{K}$ an infinite field, let $I \subset \mathbb{K}[\boldsymbol{x}]$ be an ideal which is the sum of m polynomials of degree at most d and the maximal minors of a matrix with polynomial entries also of degree at most d. We say that I is a *generic determinantal sum ideal* if the following three conditions hold:

- the ideal I is zero-dimensional and in shape position;
- the Hilbert series of $\mathbb{K}[\boldsymbol{x}]/I$ is

$$H(\tau) = \frac{\det(M(\tau^{d-1}))}{\tau^{(d-1)\binom{m-1}{2}}} \frac{(1-\tau^d)^m (1-\tau^{d-1})^{n-m}}{(1-\tau)^n}$$

  where $M(\tau)$ is the $(m-1) \times (m-1)$-matrix whose (i,j)th entry is $\sum_k \binom{m-i}{k}\binom{n-1-j}{k}\tau^k$;
- for all $e \geq 1$, the Hilbert series of $(\mathbb{K}[\boldsymbol{x}]/I)/\langle x_n^e \rangle$ is equal to the series $(1-\tau)H$ truncated at the first non-positive coefficient.

Under some regularity assumptions, the ideal defining the set of critical values of a generic polynomial map restricted to a smooth algebraic set falls in this class, see [10]. Thus, determining an asymptotic for t for such systems allows us to estimate the complexity of the Sparse-FGLM algorithm in this situation and in particular for computing the critical values of the restriction of a generic polynomial map.

Relying on manipulations of the Hilbert series given in Definition 1.8, we simplified this expression in order to show that this Hilbert series is *unimodal*, which allowed us to prove the following theorem.

Theorem 3.1 (see also [10, Th. 1])**:** Let I be a generic determinantal sum ideal of $\mathbb{K}[\boldsymbol{x}]$ as in Definition 1.8. Let $\mathcal{G}_{\text{DRL}}$ be the reduced $\prec_{\text{DRL}}$-Gröbner basis of I. Then, the latter case of Lemma 2.5 only happens if there exists $g \in \mathcal{G}_{\text{DRL}}$ such that $\text{LM}_{\prec_{\text{DRL}}}(g) = x_n\sigma_j$. As a consequence, one has $\text{NF}\left(x_n\sigma_j, \mathcal{G}_{\text{DRL}}, \prec_{\text{DRL}}\right) = x_n\sigma_j - g$.

As a consequence, the matrix of the multiplication by x_n is obtained for free from the $\prec_{\text{DRL}}$-Gröbner basis, as in Lemma 2.6 for generic systems. Taking this structure into account, we then prove that this number t is equal to the largest coefficient of the Hilbert series. Finally, using combinatorial techniques, we obtain formulae for t for such generic determinantal sum ideals.

Theorem 1.9 (see also [10, Th. 2])**:** Let I be a generic determinantal sum ideal of $\mathbb{K}[\boldsymbol{x}]$ as in Definition 1.8. Let $\mathcal{G}_{\text{DRL}}$ be the reduced $\prec_{\text{DRL}}$-Gröbner basis of I. Let t be the number of polynomials in $\mathcal{G}_{\text{DRL}}$ whose leading monomial is divisible by x_n.

Then, for $d = 2$ and $n \geq m$,

$$t = \sum_{k=0}^{m-1} \binom{n-m-1+k}{k} \binom{m}{\lfloor 3m/2 \rfloor - 1 - j}.$$

[5] INRIA
[6] former Ph.D. student at Sorbonne Université
[7] Sorbonne Université

Moreover, for $d \geq 3$ and $n \to \infty$,

$$t \approx \frac{1}{\sqrt{(n-m)\pi}} \sqrt{\frac{6}{(d-1)^2-1}} d^m (d-1)^{n-m} \binom{n-2}{m-1}.$$

In Table 3.1, we show the practical accuracy of the formulae in Theorem 1.9, for the parameter t. For $d = 2$ we use our exact formula, while for $d \geq 3$ we use the asymptotic formula. The column *Actual* is the true density of the matrix of the multiplication by x_n for $\prec_{\text{DRL}}$. The column *Theoretical* is an approximation given by the ratio t/D and the column *Asymptotic* is another one given by the asymptotic of t divided by D. The column *Actual* was already present in [53, Tab. 2], while the other two columns were added in [10]. Let us recall that in this setting, $D = d^m(d-1)^{n-m}\binom{n-1}{m-1}$, see [84, Th. 2.2].

Parameters	Degree	Matrix Density		
(d,m,n)	D	Actual	Theoretical	Asymptotic
$(2,4,9)$	896	30.17%	30.80%	30.80%
$(2,4,10)$	1344	31.13%	31.77%	31.77%
$(2,4,11)$	1920	31.86%	32.50%	32.50%
$(3,3,6)$	2160	17.52%	18.52%	27.73%
$(3,3,7)$	6480	17.39%	18.31%	26.62%
$(3,3,8)$	18144	17.63%	18.72%	25.50%
$(4,2,5)$	1728	14.46%	15.45%	21.24%
$(4,2,6)$	6480	14.11%	15.13%	19.56%
$(5,2,5)$	6400	11.00%	11.94%	15.47%
$(6,2,5)$	18000	8.80%	9.63%	12.22%

Table 3.1: Density of matrix of the multiplication by x_n for generic critical point systems

Let us also mention that during his Ph.D., Andrew Ferguson[8], with Le, extended this analysis of the asymptotic of t for ideals spanned by $(r+1)$-minors of a polynomial symmetric matrix $(f_{i,j})_{1\leq i,j\leq m}$, where the $f_{i,j}$'s are in $n = \binom{m-r+1}{2}$ variables and of degree d, see [58]. As above, their asymptotics are derived from the unimodality of the Hilbert series. As the Hilbert series is only known in the cases $r = m-2$, $r = m-1$ and $r = 1$, their asymptotics hold in these cases. For the other cases, they hold if the Hilbert series is rightfully conjectured to be unimodal.

3.1.2 A polynomial-matrix algorithm

Under the stability assumption, two algorithms, more efficient than the original FGLM algorithm, are at our disposal. The Faugère and Mou's Sparse-FGLM algorithm [52, 53], which requires that the reduced $\prec_{\text{LEX}}$-Gröbner basis is in shape position and whose complexity is in $O(tD)$ and Neiger and Schost's algorithm [83] whose complexity is in $\tilde{O}(D^\omega)$.

Despite the practical efficiency of the Sparse-FGLM algorithm and the asymptotics on t [10, 53], the recent improvements on ω, the best known bounds are $\omega \leq 2.37\,285\,96$ [1] or even $\omega \leq 2.37\,188$ [43], make Neiger and Schost's algorithm the fastest.

In this subsection, I present a change of order algorithm designed by Vincent Neiger[9], Mohab Safey El Din[9] and myself from $\prec_{\text{DRL}}$ to $\prec_{\text{LEX}}$ with complexity $\tilde{O}(t^{\omega-1}D)$, assuming that the $\prec_{\text{DRL}}$-Gröbner basis satisfies the stability assumption and that the $\prec_{\text{LEX}}$-Gröbner basis is in shape position. Since $t \leq D$, this complexity makes it asymptotically faster than the two aforementioned algorithms, independently on the value of ω and the ratio t/D.

To do so, we push forward the study of the properties of the multiplication matrix M_{x_n} by x_n. Let S_{DRL} be the monomial basis of $\mathbb{K}[\boldsymbol{x}]/I$ obtained from the reduced $\prec_{\text{DRL}}$-Gröbner basis $\mathcal{G}_{\text{DRL}}$ of I. Under the stability assumption, its columns are either unit vectors (for those $\mu \in S_{\text{DRL}}$ such that $x_n\mu \in S_{\text{DRL}}$) or vectors of coefficients of polynomials in $\mathcal{G}_{\text{DRL}}$ (for those $\mu \in S_{\text{DRL}}$ such that $x_n\mu$ is a leading monomial of one element in $\mathcal{G}_{\text{DRL}}$). The latter ones are usually referred to as a "dense"

[8] former Ph.D. student at Sorbonne Université
[9] Sorbonne Université

columns, [53, Sec. 5] and [101, Sec. 4]. This is a well-known matrix structure in $\mathbb{K}$-linear algebra, called a *shifted form* in [88, 89], and studied in particular in the context of the computation of the characteristic polynomial or the Frobenius normal form of a matrix over $\mathbb{K}$, see [102, Sec. 9.1].

Exploiting the algebraic structure itself, we relate it to operations in a $\mathbb{K}[x_n]$-submodule of I. Following a classical construction in [102, Sec. 9.1], instead of the multiplication matrix M_{x_n} which is in $\mathbb{K}^{D\times D}$, we consider a univariate polynomial matrix P_{x_n} in $\mathbb{K}[x_n]^{t\times t}$ whose average column degree is D/t. This polynomial matrix can be seen as a "compression" of M_{x_n}, or more precisely of the characteristic matrix $x_n\mathrm{Id}_D - M_{x_n}$, with smaller matrix dimension but larger degrees.

Under a less restrictive assumption than the stability one, and which is also included in the assumptions of the SPARSE-FGLM algorithm of [52, 53] and of [46], we can build P_{x_n} for free from $\mathcal{G}_{\mathrm{DRL}}$. Then, we retrieve $\mathcal{G}_{\mathrm{LEX}}$ using $\mathbb{K}[x_n]$-linear algebra operations on P_{x_n}.

Theorem 1.10 (see also [20, Th. 1.1]): Let I be a zero-dimensional ideal $\mathbb{K}[\boldsymbol{x}]$ of degree D. Let $\mathcal{G}_{\mathrm{DRL}}$ (resp. $\mathcal{G}_{\mathrm{LEX}}$) be the reduced $\prec_{\mathrm{DRL}}$- (resp. $\prec_{\mathrm{LEX}}$-)Gröbner basis of I and S_{DRL} be the $\prec_{\mathrm{DRL}}$-monomial basis of $\mathbb{K}[\boldsymbol{x}]/I$. Assume that $x_1,\ldots,x_{n-1}$ are in S_{DRL}, and that for all monomials $\mu \in S_{\mathrm{DRL}}$, either $x_n\mu$ is in S_{DRL} or it is the $\prec_{\mathrm{DRL}}$-leading monomial of an element in $\mathcal{G}_{\mathrm{DRL}}$. Assume that I is in shape position. Then, one can compute $\mathcal{G}_{\mathrm{LEX}}$ using $\tilde{O}\left(t^{\omega-1}D\right)$ operations in $\mathbb{K}$, where t is the number of elements of $\mathcal{G}_{\mathrm{DRL}}$ whose $\prec_{\mathrm{DRL}}$-leading monomial is divisible by x_n.

Example 3.2: Let $\mathcal{H}_{\mathrm{DRL}} = \left\{x_2^2 - x_1 + x_2, x_1x_2 - x_1 + x_2, x_1^2 - x_1 + x_2\right\}$ be the $\prec_{\mathrm{DRL}}$-Gröbner basis of J. Then, the associated staircase is $\{1, x_2, x_1\}$ and $D = 3$. One can read the matrix of the multiplication by x_2 in $\mathbb{K}[x_1,x_2]/J$ in this monomial basis directly from $\mathcal{H}_{\mathrm{DRL}}$. It is

$$M_{x_2} = \left(\begin{array}{ccc|c} 1 & x_2 & x_1 & \\ \hline 0 & 0 & 0 & 1 \\ 1 & -1 & -1 & x_2 \\ \hline 0 & 1 & 1 & x_1 \end{array}\right) \in \mathbb{K}^{D\times D},$$

where the ith column is the image of the ith monomial of the staircase by its multiplication by x_2.

The univariate polynomial matrix P_{x_2} is obtained from the polynomials in $\mathcal{H}_{\mathrm{DRL}}$ whose leading monomials are multiples of x_2, i.e. the first $t = 2$ polynomials of $\mathcal{H}_{\mathrm{DRL}}$, and decomposed in the $\mathbb{K}[x_2]$-module $\mathbb{K}[x_2] + x_1\mathbb{K}[x_2]$. It is

$$P_{x_2} = \begin{pmatrix} x_2^2 + x_2 & x_2 \\ -1 & x_2 - 1 \end{pmatrix}\begin{matrix} 1 \\ x_1 \end{matrix} \in \mathbb{K}[x_2]^{t\times t}.$$

Its Hermite normal form is

$$\begin{pmatrix} x_2^3 & -x_2^2 - x_2 \\ 0 & 1 \end{pmatrix}\begin{matrix} 1 \\ x_1 \end{matrix},$$

which allows us to read that the polynomials x_2^3 and $x_1 - x_2^2 - x_2$ are both in J and $\mathbb{K}[x_2] + x_1\mathbb{K}[x_2]$. Thus, they form the $\prec_{\mathrm{LEX}}$-Gröbner basis of J.

3.2 Saturated and colon ideals

Let I be an ideal of $\mathbb{K}[\boldsymbol{x}]$ and $\varphi \in \mathbb{K}[\boldsymbol{x}]$. The colon and saturation ideals of I w.r.t. φ are defined as

$$I : \langle\varphi\rangle = \{h \in \mathbb{K}[\boldsymbol{x}] \mid h\varphi \in I\}, \quad I : \langle\varphi\rangle^{\infty} = \left\{h \in \mathbb{K}[\boldsymbol{x}] \,\middle|\, \exists k \in \mathbb{N},\ h\varphi^k \in I\right\}.$$

Let us recall that by [41, Chap. 4], the algebraic set $V(I : \langle\varphi\rangle^{\infty}) \subset \overline{\mathbb{K}}^n$ is the Zariski closure of the set difference $V(I) \setminus V(\varphi)$.

Computing algebraic representations of saturated ideals arises in many applications ranging from experimental mathematics to engineering sciences (see [33, 60, 87]) since some natural algebraic modelings come with parasite solutions which one excludes through some saturation process. For instance, modeling that some $(p \times q)$-matrix with polynomial entries has rank r through the simultaneous vanishing of its $(r+1)$-minors will include those points at which the matrix has rank less than r.

Since

$$I : \langle\varphi\rangle = \frac{1}{\varphi}\left(I \cap \langle\varphi\rangle\right) = \left(tI + \langle(1-t)\varphi\rangle\right) \cap \mathbb{K}[\boldsymbol{x}]$$
$$I : \langle\varphi\rangle^{\infty} = \left(I + \langle 1 - t\varphi\rangle\right) \cap \mathbb{K}[\boldsymbol{x}],$$

using Rabinowitsch's trick [90] and [41, Chap. 4, Sec. 4, Th. 14, (ii)], one can compute Gröbner bases of these ideals choosing a monomial order eliminating t and keeping all polynomials not involving t, see also [41, Chap. 3, Sec. 1, Th. 2 and Ex. 6].

This section is about the computational problem of computing a Gröbner basis associated to $I : \langle\varphi\rangle^{\infty}$ or $I : \langle\varphi\rangle$ without introducing an extra variable. This is joint work with Christian Eder[10] and Mohab Safey El Din[11].

3.2.1 Bayer's algorithm

If I is homogeneous, i.e. it is spanned by a set of homogeneous polynomials, Bayer's algorithm [4] allows one to compute $I : \langle x_n\rangle^{\infty}$. If it is not, then one can still recover a Gröbner basis of $I : \langle x_n\rangle^{\infty}$ using Algorithm 3.1, still called Bayer's algorithm:

Input: A list of polynomials $f_1, \ldots, f_s$ spanning an ideal $I \subseteq \mathbb{K}[\boldsymbol{x}]$.
Output: A $\prec_{\text{DRL}}$-Gröbner basis of $I : \langle x_n\rangle^{\infty}$.

1 Homogenize $f_1, \ldots, f_s$ with variable x_0 to obtain $f_1^{\text{h}}, \ldots, f_s^{\text{h}}$.
2 Compute G, the $\prec_{\text{DRL}}$-Gröbner basis of $\left\langle f_1^{\text{h}}, \ldots, f_s^{\text{h}}\right\rangle$ with $x_n \prec_{\text{DRL}} \cdots \prec_{\text{DRL}} x_1 \prec_{\text{DRL}} x_0$.
3 **For each** g **in** G^{h} **do**
4 **While** $x_n \mid g$ **do** $g := g/x_n$.
5 Set x_0 to 1 in g.
6 **Return** G.

Algorithm 3.1: Bayer's algorithm

When $\varphi \neq x_n$, one introduces a slack variable x_{n+1}, computes the saturation of $I + \langle x_{n+1} - \varphi\rangle$ w.r.t. x_{n+1} and eliminate x_{n+1}.

Rabinowitsch's trick and Bayer's algorithm constitute the state-of-the-art algorithms for computing saturations of ideals. Note that they do not take advantage of intermediate data obtained during the Gröbner basis computations since these are used as black boxes.

3.2.2 F_4SAT, an algorithm for saturated ideals

After the first step of the F_4 algorithm in degree d, if no new polynomial of degree at most d is discovered, then the current Gröbner basis $\mathcal{G}$ is a d-truncated $\prec$-Gröbner basis of I. Therefore, we have a partial information on the staircase of I, and thus of $I : \langle\varphi\rangle^{\infty}$, for $\prec$ since we know monomials that are outside of this staircase. The F_4SAT algorithm searches for polynomials in $I : \langle\varphi\rangle^{\infty}$ whose supports are entirely included in the given staircase using the fact that $\left(I : \langle\varphi\rangle^{\infty}\right) : \langle\varphi\rangle = I : \langle\varphi\rangle^{\infty}$. If new polynomials are found, they are added to $\mathcal{G}$ and the necessary critical pairs are added to the set of pairs to handle. Then, we resume the F_4 algorithm.

The search of new polynomials is done through linear algebra computations. From a d-truncated Gröbner basis of an ideal J, $I \subseteq J \subseteq I : \langle\varphi\rangle^{\infty}$, we compute a bound B on the degree of the polynomials in the reduced Gröbner basis of $J : \langle\varphi\rangle^{\infty} = I : \langle\varphi\rangle^{\infty}$ using the ComputeMaxDegree subroutine based on [79, Sec. 4.5, Cor.]. Then, we compute $\text{NF}(\sigma\varphi, \mathcal{G}, \prec)$ for all monomials σ in the associated staircase S of degree at most B. Finally, we search for vanishing linear combinations thereof. Indeed, if

$$\text{NF}(s\varphi, \mathcal{G}, \prec) - \sum_{\substack{\sigma \in S_d \\ \sigma \prec s}} c_\sigma \,\text{NF}(\sigma\varphi, \mathcal{G}, \prec) = 0,$$

then $\left(s - \sum_{\sigma \in S_d, \sigma \prec s} c_\sigma \sigma\right)\varphi \in J$. This yields Algorithm 3.2.

Theorem 3.3 ([15, Th. 3.1]): Let $f_1, \ldots, f_s$ be a generating family of an ideal $I \subseteq \mathbb{K}[\boldsymbol{x}]$, $\varphi \in \mathbb{K}[\boldsymbol{x}]$ be a polynomial and $\prec$ be a total degree monomial order. Then, Algorithm 3.2 terminates and returns

[10] Rheinland-Pfälzische Technische Universität Kaiserslautern-Landau
[11] Sorbonne Université

Input: A list of polynomials $f_1, \dots, f_s$ spanning an ideal $I \subseteq \mathbb{K}[\boldsymbol{x}]$, a polynomial $\varphi \in \mathbb{K}[\boldsymbol{x}]$ and a total degree monomial order $\prec$.
Output: A $\prec$-Gröbner basis $\mathcal{G}$ of $I : \langle \varphi \rangle^\infty$.

$\mathcal{G} \coloneqq \{f_1, \dots, f_s\}$.
$b \coloneqq$ **true** // tracks if $\mathcal{G}$ has changed
$P \coloneqq \{(f_i, f_j) \mid 1 \leq i < j \leq s\}$
While $P \neq \emptyset$ **do**
  Choose a subset L of P.
  $P \coloneqq P \setminus L$.
  $L \coloneqq \text{SymbolicPreprocessing}(L, \mathcal{G})$.
  $L \coloneqq \text{LinearAlgebra}(L)$.
  For $h \in L$ *with* $LM_\prec(h) \notin \langle LM_\prec(\mathcal{G}) \rangle$ **do**
    $P \coloneqq \{(g, h) \mid g \in \mathcal{G}\}$.
    $\mathcal{G} \coloneqq \mathcal{G} \cup \{h\}$.
    $b \coloneqq$ **true**.
  If b **then** // new information on $\langle \text{LM}_\prec(I : \langle \varphi \rangle^\infty) \rangle$
    $b \coloneqq$ **false**.
    $B \coloneqq \text{ComputeMaxDegree}(\mathcal{G}, \varphi)$. // bounds the degrees in the sought Gröbner basis
    For $\sigma \notin LM_\prec(\mathcal{G})$ **and** $\deg \sigma \leq B$ **do**
      $q_\sigma \coloneqq \text{NF}(\sigma\varphi, \mathcal{G}, \prec)$.
    Build the matrix M whose rows are given by polynomials q_σ and columns by each monomials in their support in decreasing order.
    Compute a lower triangular basis K of the left-kernel of M.
    For each $k \in K$ **do**
      $h \coloneqq \sum_{\sigma \notin \langle \text{LM}_\prec(\mathcal{G}) \rangle} k_\sigma \sigma$. // the polynomial whose vector of coefficients is k
      $P \coloneqq \{(g, h) \mid g \in \mathcal{G}\}$.
      $\mathcal{G} \coloneqq \mathcal{G} \cup \{h\}$.
      $b \coloneqq$ **true**.
Return $\mathcal{G}$.

Algorithm 3.2: F_4SAT

a $\prec$-Gröbner basis of $I : \langle\varphi\rangle^\infty$.

Example 3.4: Let us call the F_4SAT algorithm on the reduced $\prec_{\text{DRL}}$-Gröbner basis $\mathcal{G}_{\text{DRL}} = \{(x_2^2 - x_1 + x_2)(x_1^2 + x_2^2 - 1), (x_1^2 - x_1x_2 - x_2^2 + x_1 - x_2)(x_1^2 + x_2^2 - 1)\}$ and $\varphi = x_1^4 - x_2^4 - 2x_1^2 + 1$ in order to compute the reduced $\prec_{\text{DRL}}$-Gröbner basis of $I : \langle\varphi\rangle^\infty$ with

$$I = \left\langle (x_2^4 + x_2^3)(x_1^2 + x_2^2 - 1), (x_1 - x_2^2 - x_2)(x_1^2 + x_2^2 - 1) \right\rangle.$$

From $\mathcal{G}_{\text{DRL}}$, we have the staircase $S_{\text{DRL}} = \{1, x_2, x_1, x_2^2, x_1x_2, x_1^2, x_2^3, x_1x_2^2, x_1^2x_2, x_1^3, x_2^4, x_1x_2^3, x_1^3x_2, x_2^5, x_1x_2^4, x_2^6, x_1x_2^5, \ldots\}$. Since this staircase is infinite, we first need a degree bound for the considered monomials in S_{DRL}. Let us choose the maximal degree in $\mathcal{G}$ minus one, i.e. 3.

The normal form of $h\varphi$ w.r.t. $\mathcal{G}$ and $\prec_{\text{DRL}}$, where h of degree 3 has its support in S_{DRL} is 0, if, and only if,

$$h = c_3(x_2^2 - x_1 + x_2) + c_5(x_1^2 - x_1x_2) + c_6(x_2^3 - x_1x_2 + x_1 - x_2) + c_7x_1x_2^2 + c_8x_1^2x_2 + c_9x_1^3.$$

Thus, we have found new polynomials in $I : \langle\varphi\rangle^\infty$. We build the ideal I', spanned by I and these polynomials, and compute its reduced $\prec_{\text{DRL}}$-Gröbner basis $\mathcal{G}'_{\text{DRL}} = \{x_2^2 - x_1 + x_2, x_1^2 - x_1x_2\}$. The new associated staircase is $S'_{\text{DRL}} = \{1, x_2, x_1, x_1x_2\}$, which is now finite.

The normal form of $h\varphi$ w.r.t. $\mathcal{G}'_{\text{DRL}}$ and $\prec_{\text{DRL}}$, where h has its support in S'_{DRL}, is 0 if, and only if,

$$h = c_3(x_1x_2 - x_1 + x_2).$$

We have found a new polynomial $x_1x_2 - x_1 + x_2$ and compute the reduced $\prec_{\text{DRL}}$-Gröbner basis $\mathcal{H}_{\text{DRL}} = \{x_2^2 - x_1 + x_2, x_1x_2 - x_1 + x_2, x_1^2 - x_1 + x_2\}$ of $J = I' + \langle x_1x_2 - x_1 + x_2\rangle$. The new associated staircase is $T_{\text{DRL}} = \{1, x_2, x_1\}$ and we cannot find a nonzero polynomial h with support in T_{DRL} whose normal form w.r.t. $\mathcal{H}_{\text{DRL}}$ and $\prec_{\text{DRL}}$ is 0. Therefore $\mathcal{H}_{\text{DRL}}$ is the reduced $\prec_{\text{DRL}}$-Gröbner basis of $I : \langle\varphi\rangle^\infty = J = \langle x_2^3, x_1 - x_2^2 - x_2\rangle$.

In Table 3.2, we present timings for different systems with coefficients in $\mathbb{Q}$. The F_4SAT algorithm is implemented in MSOLVE with a multi-modular approach using a tracer. This tracer requires two learning phases modulo two different primes before being able to run its apply phase modulo subsequent primes. We compare it with Rabinowitsch's trick implemented in MSOLVE either with probabilistic linear algebra or with the tracer, and with MAPLE and SINGULAR. Most of the time in the first learning phase is due to the last saturation step, i.e. checking in line 19 that no new polynomial are found in the saturated ideal. As we can see, the *apply* phase of F_4SAT is the fastest implementation by an order of magnitude. The SOS systems are obtained by making a polynomial f as the sum of the squares of p random polynomials of degree d. Then, we consider the ideals $\left\langle f, \frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_s} \right\rangle$ or $\left\langle \frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_s} \right\rangle$, for $s \leq n-1$, saturated by $\frac{\partial f}{\partial x_n}$.

3.2.3 SPARSE-FGLM-COLON, an algorithm for colon ideals

For an ideal $I = \langle f_1, \ldots, f_s\rangle$ and a polynomial φ, the *colon ideal* of I by φ is

$$I : \langle\varphi\rangle = \{h | h\varphi \in I\}.$$

This ideal contains I and is included in $I : \langle\varphi\rangle^\infty$ defined in Section 3.2.2. In fact, there exists an integer N such that $I : \langle\varphi^N\rangle = I : \langle\varphi^{N+1}\rangle = \cdots = I : \langle\varphi\rangle^\infty$, see [41, Chap. 4].

Let $\mathcal{G}_{\text{DRL}}$ be the reduced $\prec_{\text{DRL}}$-Gröbner basis. We assume that the colon ideal $I : \langle\varphi\rangle$ is zero-dimensional, thus $\varphi \notin I$, and that its reduced $\prec_{\text{LEX}}$-Gröbner basis $\mathcal{H}_{\text{LEX}}$ is in *shape position*: There exist $h_1, \ldots, h_n \in \mathbb{K}[x_n]$, with $\deg h_k < \deg h_n$ for $1 \leq k \leq n-1$, such that

$$\mathcal{H}_{\text{LEX}} = \{h_n(x_n), x_{n-1} - h_{n-1}(x_n), \ldots, x_1 - h_1(x_n)\}.$$

With my co-authors, Christian Eder[12] and Mohab Safey El Din[13], we designed a new algorithm in [15] for computing $\mathcal{H}_{\text{LEX}}$ from $\mathcal{G}_{\text{DRL}}$, even when I is positive-dimensional. Our approach is to build a matrix $\tilde{M}_{x_n}$ so that applying Wiedemann's algorithm allows us to recover $\mathcal{H}_{\text{LEX}}$, similarly to the SPARSE-FGLM algorithm [52, 53], see also Algorithms 2.2 and 2.3.

[12] Rheinland-Pfälzische Technische Universität Kaiserslautern-Landau
[13] Sorbonne Université

Sys-SOS	F_4SAT (learn1)	F_4SAT (learn2)	F_4SAT (apply)	msolve (prob.)	msolve (learn)	msolve (apply)	Maple (prob.)	Singular
positive-dimensional to 0-dimensional								
Steiner	115	134	67.2	204	614	153	239	3,642
d3-n6-p3	82.4	127	56.7	51.5	191	32.6	67.4	8,226
d3-n6-p4	1,592	1,776	810	2,123	5,284	1,720	3,585	–
d3-n6-p5	9,646	7,032	3,321	7,485	16,711	6,466	7,226	–
d4-n6-p2	720	1,581	536	120	520	60.6	135	24,532
d4-n6-p3	45,749	38,657	18,123	40,646	190,009	35,835	38,466	–
d2-n7-p6	18.9	41.85	10.8	31.8	101	19.5	41.4	1,773
d3-n7-p2	28.2	45.2	23.9	5.02	11.6	2.63	8.09	961
d3-n7-p3	1,462	2,688	937	953	5,851	875	1,108	–
d3-n7-p4	48,907	65,035	22,844	40,383	248,889	34,670	39,729	–
d2-n8-p4	2.68	5.04	1.89	3.55	10.1	2.02	4.12	500
d2-n8-p5	47.7	171.9	37.1	62.9	270	45.3	48.8	8,333
d2-n8-p6	287	820	169	420	1,599	301	350	54,567
d2-n8-p7	1,018	1,841	442	907	3,198	683	871	–
d3-n8-p2	300	585	266	32.4	105	20.4	50.7	9,812
d3-n8-p3	18,152	42,436	11,285	15,502	71,595	8,478	15,182	–
positive-dimensional to 1-dimensional								
d3-n6-p2	1.31	0.41	0.31	0.77	2.40	0.40	1.12	52.2
d3-n6-p3	43.7	5.55	1.84	25.2	142	16.6	35.4	2,902
d3-n6-p4	533	53.1	19.7	171	882	126	223	39,501
d3-n6-p5	1,863	184	104	276	1,145	183	394	42,854
d4-n6-p2	972	107	77	253	1,176	191	394	28,043
d4-n6-p3	31,101	1,316	596	7,444	43,803	6,336	8,817	–
d2-n7-p6	5.13	1.82	0.77	3.01	15.3	1.84	4.95	443
d3-n7-p2	13.4	3.61	2.23	9.59	54.1	5.29	12.5	872
d3-n7-p3	1,263	164	32.4	533	3,647	406	984	–
d3-n7-p4	22,296	2,235	469	6,605	47,286	5,348	10,001	–
d3-n7-p5	126,006	137,724	2,881	29,740	204,718	22,925	33,635	–
d2-n8-p5	11.7	8.37	1.79	15.1	99.9	7.92	20.4	3,972
d2-n8-p6	95.7	63.7	10.5	54.3	387	33.8	63.1	15,950
d2-n8-p7	265	79.6	22.2	81.9	556	47.2	122	15,125
d3-n8-p2	228	276	18.1	98.3	787	71.7	135	15,252
d3-n8-p3	25,593	3,716	471	11,050	107,744	8,984	13,705	–
positive-dimensional to 2-dimensional								
d2-n8-p5	620	157	43	147	663	87	166	26,746
d2-n9-p4	82	119	28	35	139	20	50	4,563
d2-n9-p5	2,889	1,617	448	924	4574	726	735	–
d2-n9-p6	40,352	8,290	2,155	8,775	40,921	7,010	6,969	–
d2-n10-p4	202	472	69	101	530	64	142	14,237
d2-n10-p5	11,347	11,512	2,801	3,373	19,465	2,710	3,148	–
positive-dimensional to 3-dimensional								
d2-n9-p4	284	100	22	75	325	46	77	10,014
d2-n9-p5	5,157	758	201	747	3,422	333	845	–
d2-n10-p4	738	481	108	180	981	109	185	28,688
d2-n10-p5	66,845	12,082	2,141	25,707	60,054	23,100	6,885	–

Table 3.2: Timings in seconds, $\prec_{\text{DRL}}$-Gröbner bases of saturated ideals.

The main ideal is twofold. First, the Sparse-FGLM algorithm builds a sequence $\boldsymbol{u}$ from $\mathcal{G}_{\text{DRL}}$, hence unless I is not Gorenstein and the initial terms of $\boldsymbol{u}$ are not random enough, $\mathrm{I_C}(\boldsymbol{u}) = I$. Then, we create a new sequence $\boldsymbol{v} = \left(\text{Eval}\left(\varphi \boldsymbol{x}^{\boldsymbol{i}}, \boldsymbol{u}\right)\right)_{\boldsymbol{i} \in \mathbb{N}^n}$ and prove that it satisfies

$$\mathrm{I_C}(\boldsymbol{v}) = \mathrm{I_C}(\boldsymbol{u}) : \langle \varphi \rangle = I : \langle \varphi \rangle .$$

Finally, it remains to recover the relations satisfied by $\boldsymbol{v}$. This is where the zero-dimensionality assumption appears, we know that only finitely many monomials m are linearly independent in $\mathbb{K}[\boldsymbol{x}]/(I : \langle \varphi \rangle)$, which is equivalent to saying that finitely many polynomials $\mathrm{NF}\left(m\varphi, \mathcal{G}_{\text{DRL}}, \prec_{\text{DRL}}\right)$ are linearly independent in $\mathbb{K}[\boldsymbol{x}]/I$. Thus, we can work in a finite-dimensional subspace W of $\mathbb{K}[\boldsymbol{x}]/I$ spanned by a set of monomials that allows us to determine $\mathcal{H}_{\text{LEX}}$.

The shape position assumption on $\mathcal{H}_{\text{LEX}}$ is of upmost importance to mimic the Sparse-FGLM algorithm by building only one matrix, related to the multiplication by x_n. Since W need not be stable by the multiplication by x_n, we consider the map multiplication by x_n composed with the projection on W. Assuming W is large enough, this new map, and its associated matrix, allow us to determine $\mathcal{H}_{\text{LEX}}$ by tweaking Algorithm 2.2 as follows in Algorithm 3.3.

Theorem 1.11 (see also [15, Th. 4.12])**:** Let I be a positive-dimensional ideal of $\mathbb{K}[\boldsymbol{x}]$, let $\mathcal{G}_{\text{DRL}}$ be its reduced $\prec_{\text{DRL}}$-Gröbner basis and S_{DRL} be the associated staircase. Let $\varphi \in \mathbb{K}[\boldsymbol{x}] \setminus I$ such that $I : \langle \varphi \rangle$ is zero-dimensional of degree D' and in shape position.

Let Σ be a finite staircase of size N containing $\operatorname{supp} \mathrm{NF}\left(x_n^i \varphi, \mathcal{G}_{\text{DRL}}, \prec_{\text{DRL}}\right)$ for all $i \in \mathbb{N}$, where $\mathrm{NF}\left(f, G, \prec\right)$ is the normal form of f w.r.t. G and $\prec$.

Let t be the number of monomials σ in Σ such that $x_n \sigma \in \mathrm{LM}_{\prec_{\text{DRL}}}(\mathcal{G}_{\text{DRL}})$ and let u be the number of monomials σ in Σ such that $x_n \sigma \in \left\langle \mathrm{LM}_{\prec_{\text{DRL}}}(I) \right\rangle \setminus \mathrm{LM}_{\prec_{\text{DRL}}}(\mathcal{G}_{\text{DRL}})$.

Then, for a generic choice of vector $\boldsymbol{r} \in \mathbb{K}^N$, Algorithm 3.3 terminates and returns the reduced $\prec_{\text{LEX}}$-Gröbner basis of $I : \langle \varphi \rangle$. To do so, it requires at most $u + n$ normal form computations w.r.t. $\mathcal{G}_{\text{DRL}}$ and $\prec_{\text{DRL}}$ plus $O\left((t + u + n) N D'\right)$ operations in $\mathbb{K}$.

As a practical optimization, we show that any zero column of the matrix $\tilde{M}$ can be removed, together with its associated row. Repeating this process can lead to considering a much smaller

Input: The reduced $\prec_{\text{DRL}}$-Gröbner basis $\mathcal{G}_{\text{DRL}}$ of a generic ideal, a polynomial $\varphi \in \mathbb{K}[\boldsymbol{x}]$, and a finite staircase Σ of size N containing $\operatorname{supp} \operatorname{NF}\left(x_n^k \varphi, \mathcal{G}_{\text{DRL}}, \prec_{\text{DRL}}\right)$ for all $k \in \mathbb{N}$.
Output: The reduced $\prec_{\text{LEX}}$-Gröbner basis of $I : \langle \varphi \rangle$, if it is in shape position.
1 Build the matrix $\tilde{M}$ of the multiplication by x_n followed by the projection on the space spanned by Σ.
2 Pick $\boldsymbol{r} \in \mathbb{K}^N$ a row-vector at random.
3 Build $\boldsymbol{\varphi}$ the column-vector of coefficients of φ restricted to Σ.
4 **For** k **from** 1 **to** $n-1$ **do**
5 Build $\boldsymbol{\psi}_k$ the column-vector of coefficients of $\operatorname{NF}(x_k\varphi, \mathcal{G}_{\text{DRL}}, \prec_{\text{DRL}})$ restricted to Σ.
6 Compute $(w_i^{(0)})_{0\le i<2N}, (w_i^{(1)})_{0\le i<N}, \ldots, (w_i^{(n-1)})_{0\le i<N})$ with Algorithm 2.2 called on $\tilde{M}, \boldsymbol{r}, \boldsymbol{\varphi}, \boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_{n-1}$.
7 $h_n \leftarrow$ Berlekamp–Massey$(w_0^{(0)}, \ldots, w_{2D-1}^{(0)})$, $D' := \deg h_n$.
8 **If** $\operatorname{NF}(h_n\varphi, \mathcal{G}_{DRL}, \prec_{DRL}) \neq 0$ **then Return** "Bad vector".
9 Compute $h_1 := \gamma_{D'-1,1}x_n^{N-1} + \cdots + \gamma_{0,1}, \ldots, h_{n-1} := \gamma_{D'-1,n-1}x_n^{N-1} + \cdots + \gamma_{0,n-1}$ with Algorithm 2.3 called on $(w_i^{(0)})_{0\le i<2D'-1}, (w_i^{(1)})_{0\le i<D'}, \ldots, (w_i^{(n-1)})_{0\le i<D'})$.
10 **For** k **from** 1 **to** $n-1$ **do**
11 **If** $\operatorname{NF}((x_k - h_k(x_n))\varphi, \mathcal{G}_{DRL}, \prec_{DRL}) \neq 0$ **then Return** "Not in shape position".
12 **Return** $\{h_n(x_n), x_{n-1} - h_{n-1}(x_n), \ldots, x_1 - h_1(x_n)\}$.

Algorithm 3.3: SPARSE-FGLM-COLON

submatrix. In Table 3.3, we give an example where a $81\,068 \times 81\,068$-matrix is replaced by a $32\,184 \times 32\,184$-submatrix for instance.

Example 3.5: Let $\mathcal{G}'_{\text{DRL}} = \left\{x_2^2 - x_1 + x_2, x_1^2 - x_1x_2\right\}$ be the $\prec_{\text{DRL}}$-Gröbner basis of the ideal I' and $\varphi = x_1^4 - x_2^4 - 2x_1^2 + 1$. We build a sequence $\boldsymbol{u}$ with random initial coefficients, e.g. $u_{0,0} = -4$, $u_{1,0} = 10$, $u_{0,1} = -3$ and $u_{1,1} = -8$, thanks to $\mathcal{G}'_{\text{DRL}}$:

$\boldsymbol{u} =$

⋮	⋮	⋮	⋮	⋮	⋱
5	5	0	0	0	⋯
4	−5	0	0	0	⋯
3	5	0	0	0	⋯
2	−13	0	0	0	⋯
1	10	−8	0	0	⋯
0	−4	−3	−8	0	⋯
i_2 \ i_1	0	1	2	3	⋯

We now consider the auxiliary sequence $\boldsymbol{v} = \left(\operatorname{Eval}\left(x_1^{i_1}x_2^{i_2}\varphi, \boldsymbol{u}\right)\right)_{(i_1,i_2)\in\mathbb{N}^2}$, i.e. $v_{i_1,i_2} = u_{i_1+4,i_2} - u_{i_1,i_2+4} - 2u_{i_1+2,j} + u_{i_1,i_2}$ for all $(i_1, i_2) \in \mathbb{N}^2$, to find the $\prec_{\text{LEX}}$-Gröbner basis $\mathcal{H}_{\text{LEX}}$ of $I' : \langle \varphi \rangle = J$,

$\boldsymbol{v} =$

⋮	⋮	⋮	⋮	⋮	⋮	⋱
4	0	0	0	0	0	⋯
3	0	0	0	0	0	⋯
2	−8	0	0	0	0	⋯
1	5	−8	0	0	0	⋯
0	17	−3	−8	0	0	⋯
i_2 \ i_1	0	1	2	3	4	⋯

By Example 2.11, we have $\mathcal{H}_{\text{LEX}} = \left\{x_2^3, x_1 - x_2^2 - x_2\right\}$.

In Table 3.3, we compare Algorithm 3.3 for computing a $\prec_{\text{LEX}}$-Gröbner basis of the zero-dimensional colon ideal $I : \langle \varphi \rangle = I : \langle \varphi \rangle^\infty$ for $\prec_{\text{LEX}}$ with MAPLE [9] using the `Groebner:-Basis` command followed by the `Groebner:-FGLM` command. The considered input generators are $I = \left\langle f, \frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_{n-1}} \right\rangle$ and $\varphi = \left(\frac{\partial f}{\partial x_n}\right)^M$ for M large enough, with f the sum of p squares of polynomials of degree d in n variables. The columns $\#\Sigma$ and $\#\Sigma'$ correspond to the size of the set Σ before and

after reductions by removing the zero columns, while column D' gives the degree of the saturated ideal. Whether it is between $\#\Sigma$ and $\#\Sigma'$ or between $\#\Sigma'$ and D', we can observe ratios going up to around 5. Therefore, it is clear that the algorithm would not be as efficient if one were to work with Σ directly. Still, it would be even more beneficial to reduce further the size of Σ' to be as close as possible to D'.

For MSOLVE, we give proportions of time spent to compute the $\prec_{\text{DRL}}$-Gröbner basis of I with F_4, to compute the normal form of correct power of φ, to build the multiplication matrix, including the computation of extra normal forms, and then to compute the $\prec_{\text{LEX}}$-Gröbner basis. We also give the time for MAPLE in seconds using Rabinowitsch's trick [90] and give the time for computing a $\prec_{\text{DRL}}$-Gröbner basis and then for the change of order step to obtain a $\prec_{\text{LEX}}$-Gröbner basis.

We can notice that the SPARSE-FGLM-COLON algorithm approach is most efficient when either the change of order step is the most time-consuming or when the ratios between $\#\Sigma$, $\#\Sigma'$ and D' are the smallest. In the former case, the algorithm benefits from the regularity of the computation of the reduced $\prec_{\text{DRL}}$-Gröbner basis of I compared to the one of $I + \langle 1 - t\varphi \rangle$ in the Rabinowitsch's trick approach. In the latter case, when Σ or Σ' are large compared to D', the overhead in the linear algebra part becomes overwhelming. Clearly, in a multi-modular approach, one would want to consider an even smaller subset of Σ' to perform the computations, once D' is known. All in all, we can see speed-ups that are significant and sometimes higher than 10.

	sizes			MSOLVE					MAPLE Groebner		
	$\#\Sigma$	$\#\Sigma'$	D'	F_4	Sat. order	Mat.	FGLM	Total	Basis	FGLM	Total
d2-n8-p5	5746	2636	1516	60%	20%	7%	13%	21	96%	4%	56
d2-n8-p6	7901	5100	3756	35%	9%	6%	50%	140	88%	12%	350
d2-n8-p7	8841	7340	6444	33%	9%	6%	52%	320	79%	21%	890
d2-n9-p5	11748	4548	2308	56%	14%	8%	22%	150	68%	32%	410
d2-n9-p6	18829	10372	6788	40%	7%	8%	45%	1200	91%	9%	3200
d2-n9-p7	24332	17540	13956	33%	5%	7%	55%	4400	83%	17%	12000
d2-n10-p4	9724	1996	652	67%	27%	5%	1%	42	99%	1%	68
d2-n10-p5	22408	7372	3340	52%	11%	9%	28%	900	97%	3%	1200
d2-n10-p6	40946	19468	11404	42%	7%	9%	42%	9900	92%	8%	17000
d3-n5-p3	3034	1320	672	39%	33%	9%	19%	1.1	97%	3%	4
d3-n5-p4	3750	2616	1968	27%	27%	11%	35%	4.3	95%	5%	43
d3-n6-p3	10773	3792	1632	60%	11%	8%	21%	50	96%	4%	77
d3-n6-p4	16271	9192	5952	37%	5%	6%	52%	500	89%	11%	1300
d3-n6-p5	18897	14862	12432	12%	5%	6%	77%	1200	82%	18%	5600
d3-n7-p3	35117	10320	3840	52%	9%	8%	31%	1300	95%	5%	1100
d3-n7-p4	62104	29760	16800	19%	4%	11%	66%	12000	91%	9%	31000
d4-n5-p3	15881	7560	4104	41%	6%	5%	48%	200	94%	6%	620
d4-n5-p4	19274	14088	11016	32%	4%	4%	60%	1000	86%	14%	4700
d4-n6-p2	41189	8424	1944	74%	5%	9%	12%	590	97%	3%	224
d4-n6-p3	81068	32184	14904	16%	3%	12%	69%	11000	92%	8%	27000
d5-n4-p3	7235	4540	3040	13%	24%	11%	52%	9	93%	7%	180
d5-n5-p3	54787	27360	15360	8%	4%	7%	81%	5100	92%	8%	22000

Table 3.3: Timings in seconds, $\prec_{\text{LEX}}$-Gröbner bases, positive-to-0-dimensional case.

3.3 Parametrizations of the radical

Proposition 3.6 (see also [15, Prop. 4.17])**:** Let I be a positive-dimensional ideal of $\mathbb{K}[\boldsymbol{x}]$, let $\mathcal{G}_{\text{DRL}}$ be its reduced $\prec_{\text{DRL}}$-Gröbner basis and S_{DRL} be the associated staircase. Let $\varphi \in \mathbb{K}[\boldsymbol{x}] \setminus I$ such that $I : \langle \varphi \rangle$ is zero-dimensional, let $\mathcal{H}_{\text{LEX}}$ be its reduced $\prec_{\text{LEX}}$-Gröbner basis and let $h_n \in \mathcal{H}_{\text{LEX}} \cap \mathbb{K}[x_n]$.

If $\sqrt{I : \langle \varphi \rangle}$ is in shape position, then one can compute its reduced $\prec_{\text{LEX}}$-Gröbner basis calling the SPARSE-FGLM-COLON algorithm with the following modifications:

(1) On line 7, h_n is the squarefree part of the polynomial returned by the Berlekamp–Massey algorithm.

(2) On line 9, h_k is obtained thanks to [66, Algo. 2], see also [29].

In practice, when an ideal J is not in shape position, it is not easy to check that $\sqrt{J}$ is. Therefore, using the following lemma is the cornerstone of our probabilistic verification algorithm in MSOLVE [13, 14] when J is not in shape position but its radical might be, see [13, Sec. 4.4].

Lemma 1.12 (see also [15, Lem. 4.15])**:** Let I be a zero-dimensional ideal of $\mathbb{K}[\boldsymbol{x}]$. Let $\lambda \in \bar{\mathbb{K}}$ be generic. Then, for $1 \leq k \leq n$, $I = I : \langle x_k + \lambda \rangle$.

We proceed as follows.

(1) Compute the polynomials $x_k - g_k(x_n)$ in $\sqrt{J}$ for $1 \leq k \leq n-1$, with $\deg g_k$ minimal.

(2) Compute the polynomials $x_k - g'_k(x_n)$ in $\sqrt{J : \langle x_k + \lambda \rangle}$ for λ picked at random and $1 \leq k \leq n-1$,

with $\deg g'_k$ minimal.

(3) Check whether $g_k = g'_k$ for $1 \leq k \leq n-1$.

By Lemma 1.12, for a generic λ, $J = J : \langle x_k + \lambda \rangle$, hence both radical ideals are the same. Furthermore, if they are in shape position, then $g_k = g'_k$ for $1 \leq k \leq n-1$. Therefore, any discrepancy must come from the fact that $\sqrt{J}$ is *not* in shape position and the polynomials $x_k - g_k(x_n)$ and $x_k - g'_k(x_n)$ are meaningless.

Chapter 4

Guessing Gröbner bases

This chapter is devoted to guessing Gröbner bases of ideals of C-relations of sequences. That is, given a finite number of terms of a sequence, the goal is to discover the linear recurrence relations, given by polynomials, satisfied by these few terms.

In some applications, like the Gröbner bases change of order one, see Sections 2.1.2 and 3.1, or in coding theory [23, 65, 96], a bound on the sizes of the relations is known. Therefore, testing the relations with sufficiently enough terms ensures that these are satisfied by the whole sequence. In other cases, like in combinatorics or number theory [3, 25, 30, 31] no a priori bound is known. In which case, one must prove afterwards the validity of the relations.

We first recall how to guess C-relations for uni-indexed sequences using the seminal Berlekamp–Massey algorithm [8, 77] and its faster variants [34] in Section 4.1. Then, we describe extensions of this algorithm to multi-indexed sequences, where my contributions lie, in Section 4.2.

This chapter is based on joint work with Brice Boyer[1], Jean-Charles Faugère[2,3] and Mohab Safey El Din[4] [11, 12, 17, 19, 21].

In all this chapter, *multi-Hankel* matrices will be used. These generalize the notion of *Hankel* matrices to multi-indexed as follows: given two sets of monomials T_1 and T_2 in $\boldsymbol{x}$ and a sequence $\boldsymbol{u}$,

$$H_{T_1,T_2} = \begin{array}{c} \\ \boldsymbol{x}^i \in T_1 \end{array} \overset{\begin{array}{ccc} \cdots & \boldsymbol{x}^j \in T_2 & \cdots \end{array}}{\begin{pmatrix} \ddots & \vdots & \iddots \\ \cdots & \operatorname{Eval}\left(\boldsymbol{x}^i \boldsymbol{x}^j, \boldsymbol{u}\right) & \cdots \\ \iddots & \vdots & \ddots \end{pmatrix}}$$

4.1 Uni-indexed sequences and the Berlekamp–Massey algorithm

Let $\boldsymbol{u} = (u_i)_{i \in \mathbb{N}}$ be a one-dimensional sequence that we assume C-finite. The goal is to find the unique polynomial $G_{x^d} = x^d + \gamma_{x^{d-1}} x^{d-1} + \cdots + \gamma_1$ with d minimal such that $\mathrm{I_C}(\boldsymbol{u}) = \langle G_{x^d} \rangle$. Though, in general d is not known and only the first N terms of $\boldsymbol{u}$: $u_0, \ldots, u_{N-1}$ are known. Therefore, we aim to guess G_{x^d} using the fact that it must satisfies

$$\operatorname{Eval}\left(G_{x^d}, \boldsymbol{u}\right) = \cdots = \operatorname{Eval}\left(G_{x^d} x^{N-d-1}, \boldsymbol{u}\right) = 0. \tag{4.1}$$

4.1.1 Matrix viewpoints

Requiring that d is minimal and $\gamma_1, \ldots, \gamma_{x^{d-1}}$ are such that Equation (4.1) is satisfied is equivalent to looking for the largest collection of vectors $\begin{pmatrix} \gamma_1 \\ \vdots \\ \gamma_{x^{d-1}} \\ 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ \gamma_1 \\ \vdots \\ \gamma_{x^{d-1}} \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \ldots, \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ \gamma_1 \\ \vdots \\ \gamma_{x^{d-1}} \\ 1 \end{pmatrix}$ in the kernel of

$$H_{\{1\},\{1,\ldots,x^{d-1},x^d,x^{d+1},\ldots,x^{N-1}\}} = \begin{array}{c} 1 \\ \end{array} \overset{\begin{array}{ccccccc} 1 & \cdots & x^{d-1} & x^d & x^{d+1} & \cdots & x^{N-1} \end{array}}{\begin{pmatrix} u_0 & \cdots & u_{d-1} & u_d & u_{d+1} & \cdots & u_{N-1} \end{pmatrix}}.$$

[1] former Post-doc at Sorbonne Université
[2] INRIA
[3] CryptoNext Security
[4] Sorbonne Université

Equivalently, one also looks for the smallest integer d such that

$$H_{\mathcal{T}_{\leq x^{N-d-1}},\mathcal{T}_{\leq x^d}}\begin{pmatrix}\gamma_1\\ \vdots\\ \gamma_{x^{d-1}}\\ 1\end{pmatrix} = \begin{array}{c} \\ 1\\ x\\ \vdots\\ x^{N-d-1}\end{array}\begin{array}{c}\begin{array}{cccc} 1 & \cdots & x^{d-1} & x^d\end{array}\\ \begin{pmatrix} u_0 & \cdots & u_{d-1} & u_d\\ u_1 & \cdots & u_d & u_{d+1}\\ \vdots & & \vdots & \vdots\\ u_{N-d-1} & \cdots & u_{N-2} & u_{N-1}\end{pmatrix}\end{array}\begin{pmatrix}\gamma_1\\ \vdots\\ \gamma_{x^{d-1}}\\ 1\end{pmatrix} = 0,$$

where $\mathcal{T}_{\leq m}$ denotes the set of all monomials less or equal to m for $\prec$, the only monomial order on $\mathbb{K}[x]$.

With this modeling of the problem, we are led to a polynomial viewpoint.

4.1.2 Polynomial viewpoint

The Hankel matrix-vector product can be extended into

$$\begin{pmatrix} u_0 & \cdots & u_{d-1} & u_d\\ u_1 & \cdots & u_d & u_{d+1}\\ \vdots & & \vdots & \vdots\\ u_{N-d-1} & \cdots & u_{N-2} & u_{N-1}\\ u_{N-d} & \cdots & u_{N-1} & 0\\ \vdots & \iddots & \iddots & \vdots\\ u_{N-1} & 0 & \cdots & 0\end{pmatrix}\begin{pmatrix}\gamma_1\\ \vdots\\ \gamma_{x^{d-1}}\\ 1\end{pmatrix} = \begin{pmatrix}0\\ 0\\ \vdots\\ 0\\ r_{x^{d-1}}\\ \vdots\\ r_1\end{pmatrix}, \tag{4.2}$$

representing the product of the two polynomials $P_{\mathcal{T}_{\leq x^{N-1}}} = \sum_{i=0}^{N-1} u_i x^{N-i-1}$, given by the sequence terms, and $G_{x^d} = x^d + \sum_{k=0}^{d-1}\gamma_{x^k}x^k$, giving the C-relation, modulo $B = x^N$. This can rewritten as follows, there exists $Q \in \mathbb{K}[x]$ such that

$$Q\cdot x^N + G_{x^d}\sum_{i=0}^{N-1} u_i x^{N-i-1} = R, \quad \deg R < d, \tag{4.3}$$

which is a Bézout relation between x^N and $P_{\mathcal{T}_{\leq x^{N-1}}}$, where we ask that the degree of the right-hand side member, or equivalently its leading term, is less than that of the cofactor of $P_{\mathcal{T}_{\leq x^{N-1}}}$.

The main advantage of this viewpoint is that we can now compute d and G_{x^d} using the extended Euclidean algorithm on x^N and $P_{\mathcal{T}_{\leq x^{N-1}}}$ and stop the algorithm when a Bézout relation satisfies Equation (4.3).

This viewpoint gives rise to Algorithm 4.1, which is a variant of the BM algorithm and a modified extended Euclidean algorithm. While, as written, its complexity is quadratic in N, one can use fast Euclidean algorithm [34] to improve the time complexity to $O(\mathsf{M}(n)\log n)$.

Input: $u_0,\ldots,u_{N-1}$, the first N terms of a sequence $\boldsymbol{u}$.
Output: A polynomial C of degree d, with d minimal such that $\mathrm{Eval}(Cx^i,\boldsymbol{u}) = 0$ for all $0 \leq i \leq N-d-1$.

$m := {}^1\!/_x$, $L_m := [x^N, 0]$.
$m' := 1$, $L_{m'} := \left[\sum_{i=0}^{N-1} u_i x^{N-i-1}, 1\right]$.
While $\deg L_{m',2} \leq \deg L_{m',1}$ **do**
  $Q := \mathsf{Quo}(L_{m,1}, L_{m',1})$.
  $L_{m'\,\mathrm{LM}_\prec(Q)} := L_m - QL_{m'}$.
  $m, m' := m', m'\,\mathrm{LM}_\prec(Q)$.
Return $L_{m',2}$.

Algorithm 4.1: Berlekamp–Massey/ Extended Euclidean algorithm

Example 4.1: Let us consider the sequence $\boldsymbol{w} = (w_i)_{i\in\mathbb{N}}$,

$\boldsymbol{w} =$

i	0	1	2	3	4	5	6	7	8	$\cdots$
	-4	10	-13	5	-5	5	-5	5	-5	$\cdots$

and assume $N = 9$.

With the matrix viewpoint, the kernel of

$$H_{\{1\},\{1,x,x^2,x^3,x^4,x^5,x^6,x^7,x^8\}} = \begin{array}{c} \\ 1 \end{array}\begin{array}{c} \begin{array}{ccccccccc} 1 & x & x^2 & x^3 & x^4 & x^5 & x^6 & x^7 & x^8 \end{array} \\ \begin{pmatrix} -4 & 10 & -13 & 5 & -5 & 5 & -5 & 5 & -5 \end{pmatrix} \end{array},$$

gives us the collection of vectors $\begin{pmatrix}0\\0\\0\\1\\1\\0\\0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\0\\0\\1\\1\\0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\0\\0\\0\\1\\1\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\0\\0\\0\\0\\1\\1\\0\end{pmatrix}, \begin{pmatrix}0\\0\\0\\0\\0\\0\\0\\1\\1\end{pmatrix}$, hence $\mathrm{I_C}(\boldsymbol{w}) = \langle x^4 + x^3 \rangle$.

Equivalently, we have that

$$H_{\{1,x,x^2,x^3,x^4\},\{1,x,x^2,x^3,x^4\}} \begin{pmatrix}0\\0\\0\\1\\1\end{pmatrix} = \begin{array}{c} \\ 1 \\ x \\ x^2 \\ x^3 \\ x^4 \end{array}\begin{array}{c} \begin{array}{ccccc} 1 & x & x^2 & x^3 & x^4 \end{array} \\ \begin{pmatrix} -4 & 10 & -13 & 5 & -5 \\ 10 & -13 & 5 & -5 & 5 \\ -13 & 5 & -5 & 5 & -5 \\ 5 & -5 & 5 & -5 & 5 \\ -5 & 5 & -5 & 5 & -5 \end{pmatrix} \end{array} \begin{pmatrix}0\\0\\0\\1\\1\end{pmatrix} = 0$$

but

$$H_{\{1,x,x^2,x^3,x^4,x^5\},\{1,x,x^2,x^3\}} = \begin{array}{c} \\ 1 \\ x \\ x^2 \\ x^3 \\ x^4 \\ x^5 \end{array}\begin{array}{c} \begin{array}{cccc} 1 & x & x^2 & x^3 \end{array} \\ \begin{pmatrix} -4 & 10 & -13 & 5 \\ 10 & -13 & 5 & -5 \\ -13 & 5 & -5 & 5 \\ 5 & -5 & 5 & -5 \\ -5 & 5 & -5 & 5 \\ 5 & -5 & 5 & -5 \end{pmatrix} \end{array}$$

has full rank, ensuring that no relation given by a polynomial of degree 3 exists.

With the polynomial viewpoint, we perform the extended Euclidean algorithm on x^9 and $P_{\mathcal{T}_{\leq x^8}}$ and find

$$\begin{aligned}
L_{1/x} &= [x^9, 0] \\
L_1 &= [-4x^8 + 10x^7 - 13x^6 + 5x^5 - 5x^4 + 5x^3 - 5x^2 + 5x - 5, 1] \\
L_x &= [24x^7 - 55x^6 + 15x^5 - 15x^4 + 15x^3 - 15x^2 + 15x - 25, 2x + 5] \\
L_{x^2} &= [-1\,237x^6 + 285x^5 - 285x^4 + 285x^3 - 285x^2 + 45x - 595, 48x^2 + 110x + 119] \\
L_{x^3} &= [-2\,560x^5 + 2\,560x^4 - 2\,560x^3 - 3\,625x^2 + 2\,200x - 1\,600, \\
&\qquad 1\,237x^3 + 285x^2 - 120x + 320] \\
L_{x^4} &= [-5x^3, x^4 + x^3] = [R_{x^4}, G_{x^4}],
\end{aligned}$$

and $G_{x^4} = x^4 + x^3$ is a valid relation.

Remark 4.2: The BM algorithm always returns a non-zero relation. If no pair $L_{x^\delta} = [R_{x^\delta}, G_{x^\delta}]$ satisfies the requirements, then it will return a pair L_{x^d} with $\mathrm{LM}_{\prec}(G_{x^d}) \succ x^{N-1}$. From a matrix viewpoint, it returns an element of the kernel of the empty matrix $H_{\emptyset,\mathcal{T}_{\leq x^d}}$.

4.2 Extensions to multi-indexed sequences

This section is devoted to several extensions of the uni-indexed case to the multi-indexed one. It is based on several joint works with Brice Boyer[5], Jean-Charles Faugère[6,7] and Mohab Safey El Din[8].

A first extension to the BM algorithm was proposed by Sakata for bi-indexed sequences [93] and then was extended to n-indexed sequences [95, 97]. This is the so-called Berlekamp–Massey–Sakata (BMS) algorithm. Then, Brice Boyer[5], Jean-Charles Faugère[6,7] and I designed a linear-algebra-based one, called Scalar-FGLM, in [11, 12] and an adaptive variant thereof. Mourrain also proposed another algorithm using a Gram–Schmidt process in [81], called AGbb. Finally Jean-Charles Faugère[6,7] and I on one hand and Mohab Safey El Din[8] and I on another hand proposed several extensions and improvements of the Scalar-FGLM algorithm in [17, 19, 21].

We assume now that $\prec$ is a monomial order such that for any monomial $\boldsymbol{x}^{\boldsymbol{i}} \in \mathbb{N}^n$, the set $\mathcal{T}_{\preceq \boldsymbol{x}^{\boldsymbol{i}}}$ is finite. Such orders compare first monomials by (weighted) degrees, like $\prec_{\text{DRL}}$. While $\prec_{\text{LEX}}$ is not such a monomial order, one can choose a monomial order $\prec$ such that the $\prec$-Gröbner basis and the $\prec_{\text{LEX}}$-Gröbner basis coincide by putting large weights on the first variables. Though, these weights will depend on the ideal of C-relations one is dealing with.

4.2.1 The BMS algorithm

Given a multi-indexed sequence $\boldsymbol{u} = (u_{\boldsymbol{i}})_{\boldsymbol{i}\in\mathbb{N}^n}$, a monomial order $\prec$ and a monomial $M = x_1^{a_1}\cdots x_n^{a_n}$, the BMS algorithm extends the BM algorithm by computing vectors in the kernel of a multi-Hankel matrix

$$H_{\{1\},\mathcal{T}_{\preceq M}} = \begin{array}{c} \\ 1 \\ \\ \end{array}\begin{array}{c} \begin{array}{ccc} 1 & \cdots & M \end{array} \\ \begin{pmatrix} u_{0,\ldots,0} & \cdots & u_{a_1,\ldots,a_n} \end{pmatrix} \end{array},$$

corresponding to having relations $\text{Eval}\left(G_m\boldsymbol{x}^{\boldsymbol{i}},\boldsymbol{u}\right) = 0$, with $\text{LM}_{\prec}(G_m) = m$ minimal for the division and for all $\boldsymbol{x}^{\boldsymbol{i}}$ such that $m\boldsymbol{x}^{\boldsymbol{i}} \preceq M$.

While the BMS algorithm computes the C-relations using polynomial multiplications by monomials and additions, Jean-Charles Faugère[6,7] and I proposed in [17, 19] the following polynomial interpretation. This is a key argument towards the design of our algorithm that guesses the C-relations using polynomial arithmetic, including multivariate polynomial reductions, which extends the univariate Euclidean division.

Guessing a $\prec$-Gröbner basis of $\text{I}_{\text{C}}(\boldsymbol{u})$ comes down to finding the least (for the partial order $|$) monomials $m_1,\ldots,m_r \preceq M$ such that $\dim\ker H_{\mathcal{T}_{\preceq s_k},\mathcal{T}_{\preceq m_k}} > 0$ with s_k the greatest monomial such that $s_k m_k \preceq M$ for all k, $1 \le k \le r$. Indeed, by the minimality of the leading monomials, if sufficiently many terms of $\boldsymbol{u}$ are known, these polynomials form a minimal $\prec$-Gröbner basis of $\text{I}_{\text{C}}(\boldsymbol{u})$.

Then, each multi-Hankel matrix-vector product can be extended further as in equation (4.2), taking the multi-Hankel matrix $H_{\mathcal{T}_{\preceq M},\mathcal{T}_{\preceq m_k}}$ and replacing by zero any sequence term $u_{\boldsymbol{i}+\boldsymbol{j}} = \text{Eval}\left(\boldsymbol{x}^{\boldsymbol{i}}\boldsymbol{x}^{\boldsymbol{j}},\boldsymbol{u}\right)$ that we do not know, i.e. whenever $\boldsymbol{x}^{\boldsymbol{i}}\boldsymbol{x}^{\boldsymbol{j}} \succ M$. This yields the linear system

$$\begin{pmatrix} \text{Eval}\left(1,\boldsymbol{u}\right) & \cdots & \text{Eval}\left(m_k^-,\boldsymbol{u}\right) & \text{Eval}\left(m_k,\boldsymbol{u}\right) \\ \text{Eval}\left(1^+,\boldsymbol{u}\right) & \cdots & \text{Eval}\left(1^+m_k^-,\boldsymbol{u}\right) & \text{Eval}\left(1^+m_k,\boldsymbol{u}\right) \\ \vdots & & \vdots & \vdots \\ \text{Eval}\left(s_k,\boldsymbol{u}\right) & \cdots & \text{Eval}\left(s_k m_k^-,\boldsymbol{u}\right) & \text{Eval}\left(s_k m_k,\boldsymbol{u}\right) \\ \text{Eval}\left(s_k^+,\boldsymbol{u}\right) & \cdots & \text{Eval}\left(s_k^+ m_k^-,\boldsymbol{u}\right) & 0 \\ \vdots & & \iddots & \vdots \\ \text{Eval}\left(M,\boldsymbol{u}\right) & 0 & \cdots & 0 \end{pmatrix} \begin{pmatrix} \gamma_1 \\ \vdots \\ \gamma_{m_k^-} \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ f_{\mathcal{M}/s_k^+} \\ \vdots \\ f_{\mathcal{M}/a} \end{pmatrix}, \tag{4.4}$$

where $\mathcal{M} = \text{LCM}(\mathcal{T}_{\preceq M}) = x_1^{N_1}\cdots x_n^{N_n}$ and where, for a monomial μ, μ^- (resp. μ^+) is its preceding

[5] former Post-doc at Sorbonne Université
[6] INRIA
[7] CryptoNext Security
[8] Sorbonne Université

(resp. following) monomial for $\prec$. Let us notice that $\mathrm{Eval}\left(s_k^+ m_k^-, \boldsymbol{u}\right)$ can also be a 0 if $s_k^+ m_k^- > M$ and that, more generally, the gray zeroes need not be diagonally aligned like they are in the univariate case.

Such product represents the product of the polynomials $P_{\mathcal{T}_{\preceq M}} = \sum_{\boldsymbol{x}^i \preceq M} u_i \frac{M}{\boldsymbol{x}^i}$ and $G_{m_k} = m_k + \sum_{\boldsymbol{x}^i \prec m_k} \gamma_{\boldsymbol{x}^i} \boldsymbol{x}^i$ modulo $B = (x_1^{N_1}, \ldots, x_n^{N_n})$. The requirement for G_{m_k} to encode a valid relation is now that $\mathrm{LM}_\prec(R_{m_k}) \prec \frac{M}{s_k}$ with $R_{m_k} = P_{\mathcal{T}_{\preceq M}} G_{m_k} \bmod B$. To compute these relations, one start with the pairs $L_{1/x_i} = [x_i^{N_i}, 0]$ for all $1 \leq i \leq n$ and $L_1 = [P_{\mathcal{T}_{\preceq M}}, 1]$. Then, one computes S-polynomials and polynomial reductions on the first coordinates of the pairs, and report the operations on the second coordinates of the pairs until finding all pairs $L_m = [R_m, G_m]$, where the leading monomial of R_m is small enough for G_m to encode a valid relation, for all m minimal for the partial order $|$.

Theorem 1.13 (see also [19, Th. 1])**:** Let $\boldsymbol{u}$ be a sequence, $\prec$ be a weighted degree monomial order and M be a monomial. Let us assume that the reduced $\prec$-Gröbner basis $\mathcal{G}$ of $\mathrm{I_C}(\boldsymbol{u})$ and its associated staircase S satisfy $\max(S \cup \mathrm{LM}_\prec(\mathcal{G})) \preceq M$ and for all $m \preceq M$, $s = \max_{\sigma \preceq M} \{\sigma \mid \sigma m \preceq M\}$, we have $\max(S) \preceq s$. Then, the variant of the Berlekamp–Massey–Sakata algorithm using polynomial arithmetic terminates and computes $\mathcal{G}$ in $O\left(\#S(\#S + \#\mathcal{G})\#\mathcal{T}_{\preceq M}\right)$ operations in the base field, where $\mathcal{T}_{\preceq M}$ is the set of monomials less or equal to M.

Remark 4.3: As for the BM algorithm, the BMS algorithm will always return a relation G_m with $\mathrm{LM}_\prec(G_m) = m$ a pure power in each variable. Therefore, it can return G_m with $m > M$, corresponding to a vector in the kernel of the empty matrix $H_{\emptyset, \mathcal{T}_{\preceq m}}$.

4.2.2 The Scalar-FGLM algorithm

In [11, 12], Brice Boyer[9], Jean-Charles Faugère[10,11] and I designed an algorithm, Scalar-FGLM, that computes the reduced $\prec$-Gröbner basis of $\mathrm{I_C}(\boldsymbol{u})$ by means of linear algebra. Together with $\boldsymbol{u}$, it takes as an input a set of monomials T, ordered for $\prec$, and it computes the right kernel of the multi-Hankel matrix $H_{T,T}$. Vectors in this kernel can be seen as polynomials in $\mathbb{K}[\boldsymbol{x}]$ and these polynomials with a leading term minimal for the partial order induced by the division are the ones returned by the algorithm. Furthermore, if T contains the staircase and the leading monomials of the reduced $\prec$-Gröbner basis of $\mathrm{I_C}(\boldsymbol{u})$, then the Scalar-FGLM algorithm returns this Gröbner basis.

Input: A sequence $\boldsymbol{u} = (u_i)_{i \in \mathbb{N}^n}$ with coefficients in $\mathbb{K}$, a monomial ordering $\prec$, a sufficiently large staircase T and ordered for $\prec$.
Output: A reduced Gröbner basis of the ideal of C-relations of $\boldsymbol{u}$.

Build the matrix $H_{T,T}$.
Compute the set $S \subseteq T$ of smallest monomials, for $\prec$, such that $\mathrm{rank}\, H_{S,S} = \mathrm{rank}\, H_{T,T}$.
For all $m \in T \setminus S$ **do** // stabilize S for the division
  If $\exists s \in S$ *such that* $m \mid s$ **then** $S := S \cup \{m\}$.
$L := T \setminus S$ sorted for $\prec$.
$G := \emptyset$.
While $L \neq \emptyset$ **do**
  $m := \min_\prec L$
  Solve the linear system $H_{S,S}\boldsymbol{\gamma} + H_{S,\{g\}} = 0$.
  $G := G \cup \{m + \sum_{s \in S} \gamma_s s\}$.
  Remove m and any of its multiples from L.
Return G.

Algorithm 4.1: Scalar-FGLM

The algorithm computes the column rank profile of the matrix $H_{T,T}$, that is the set of leftmost linearly independent columns of the matrix. Since these columns are independent from the previous

[9] former Post-doc at Sorbonne Université
[10] INRIA
[11] CryptoNext Security

ones, their labels cannot be the leading monomial, for $\prec$, of any polynomial in the ideal of C-relations, thus they are in the associated staircase of the reduced $\prec$-Gröbner basis of this ideal.

Example 4.4: Let

$$v = \begin{array}{|c|c|c|c|c|c|c|}\hline \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \iddots \\ \hline 4 & 0 & 0 & 0 & 0 & 0 & \cdots \\ \hline 3 & 0 & 0 & 0 & 0 & 0 & \cdots \\ \hline 2 & -8 & 0 & 0 & 0 & 0 & \cdots \\ \hline 1 & 5 & -8 & 0 & 0 & 0 & \cdots \\ \hline 0 & 17 & -3 & -8 & 0 & 0 & \cdots \\ \hline i_2 \diagdown i_1 & 0 & 1 & 2 & 3 & 4 & \cdots \\ \hline \end{array}$$

and assume we want to guess its reduced $\prec_{\text{DRL}}$-Gröbner basis knowing that it contains no polynomial of degree higher than 3. We build the multi-Hankel $H_{T,T}$ where T contains all the monomials of degree at most 3 and then determine its column-rank-profile, i.e. the leftmost independent columns,

$$H_{T,T} = \begin{array}{c} \\ 1 \\ y \\ x \\ y^2 \\ xy \\ x^2 \\ y^3 \\ xy^2 \\ x^2y \\ x^3 \end{array} \begin{array}{c} \begin{array}{cccccccccc} 1 & y & x & y^2 & xy & x^2 & y^3 & xy^2 & xy^2 & x^3 \end{array} \\ \begin{pmatrix} 17 & 5 & -3 & -8 & -8 & -8 & 0 & 0 & 0 & 0 \\ 5 & -8 & -8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -3 & -8 & -8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}.$$

Clearly, the first 3 columns are independent, then the other columns, labeled with $y^2, xy, \ldots$ linearly depend on them. Thus, $\{1, y, x\} = S_{\text{DRL}}$. The leading monomials of the reduced $\prec_{\text{DRL}}$-Gröbner basis are y^2, xy and x^2 and we find the polynomials $y^2 - x + y$, $xy - x + y$ and $x^2 - x + y$.

The polynomial viewpoint presented in Section 4.2.1 for the BMS algorithm is in fact more general. Taking the same notation as in Section 4.2.1, consider the more subtle condition for a relation to be valid: the leading monomial of R_m, after discarding some monomials, is sufficiently small. Then, the polynomial viewpoint allows one to compute vectors in the kernel of a more general multi-Hankel, as those encountered in the Scalar-FGLM algorithm. We refer the reader to [17, 19] for more details.

When the nonzero terms of the sequence all lie in a cone $\mathscr{C}$, we can decide to restrict the construction of the multi-Hankel matrix to monomials in the associated algebra, i.e. the set of monomials $\mathscr{T}(\mathscr{C}) = \{\boldsymbol{x}^{\boldsymbol{i}} \mid \boldsymbol{i} \in \mathscr{C}\}$. In that setting, instead of guessing classical $\prec$-Gröbner bases, the guessed polynomials form a $\prec$-sparse Gröbner basis, as defined in [7, 56].

Theorem 4.5 (see also [21, Th. 3.2])**:** Let $\mathscr{C}$ be cone which is also a submonoid of $\mathbb{N}^n$. Let $\prec$ be a monomial order on the monomials in $\boldsymbol{x}$. Let $T \subseteq \mathscr{T}(\mathscr{C})$ be a set of monomials stable by division.

Let $\boldsymbol{u}$ be a C-finite sequence such that $\mathrm{I}_{\mathrm{C}}(\boldsymbol{u})$ has a reduced $\prec$-sparse Gröbner basis $\mathscr{G}$ with support in T. Then, the Scalar-FGLM algorithm called on $\boldsymbol{u}$, T and $\prec$ computes $\mathscr{G}$.

4.2.3 Adaptive algorithms

In many applications, including the change of order one when the ideal is not in shape position, computing sequence terms is expensive so that building such a large multi-Hankel matrix is prohibitive. In [11, 12], my co-authors and I proposed an adaptive algorithm that starts with a 1×1-matrix given by $T = \{1\}$. Then, it makes it grow by adding a monomial in T. If the rank of the matrix grows, then we know that this new monomial is in the staircase of the sought Gröbner basis. Otherwise, we have guessed a polynomial in the target Gröbner basis.

This approach is similar to the original FGLM algorithm, though we might find *fake* C-relations when the first terms of the sequence are not generic.

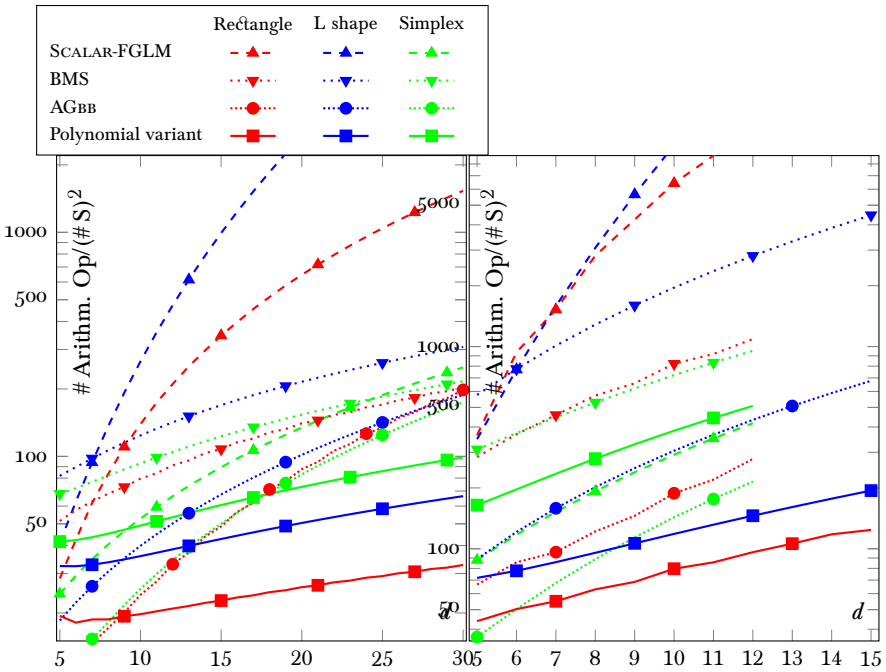

Figure 4.1: Numbers of arithmetic operations (2D/3D)

Later, Jean-Charles Faugère[12,13] and I designed an adaptive variant of the BMS algorithm in [18] which can avoid testing some candidate relations, and thus requiring some sequence terms, when a bound of the degree of the zero-dimensional ideal of C-relations is known. Furthermore, in [19], we used the polynomial viewpoint of the Scalar-FGLM algorithm of [17] and Section 4.2.2 to have an adaptive algorithm using only multivariate polynomial arithmetic. In particular, we prove the following theorem.

Theorem 1.14 (see also [19, Th. 26])**:** Let $\boldsymbol{u}$ be a sequence whose ideal of C-relations is zero-dimensional. Let $\prec$ be a monomial order, $\mathcal{G}$ be the reduced $\prec$-Gröbner basis of $\mathrm{I_C}(\boldsymbol{u})$ and S be the associated staircase.

Assuming the Adaptive Scalar-FGLM algorithm called on $\boldsymbol{u}$ returns a Gröbner basis $\mathcal{G}'$ with staircase S' and $\#S' = \#S$, then $S' = S$ and $\mathcal{G}' = \mathcal{G}$. Furthermore, the algorithm does not need more that $\#2(S \cup \mathrm{LM}_\prec(\mathcal{G}))$ sequence queries and $O((\#S+\#\mathcal{G})^2 \#2S)$ operations to recover $\mathcal{G}$, where $2S$ is the Minkowski sum of S with itself.

4.3 Benchmarks

In Figure 4.1, we consider three families of $\prec_{\mathrm{DRL}}$-Gröbner basis based on the shape of their staircase:

Rectangle: $\mathrm{LM}_\prec(\mathcal{G}) = \left\{y^{\lfloor d/2 \rfloor}, x^d\right\}$ in dimension 2 and $\mathrm{LM}_\prec(\mathcal{G}) = \left\{z^{\lceil d/3 \rceil}, y^{\lfloor d/2 \rfloor}, x^d\right\}$ dimension 3. This case is the best for the size of the Gröbner basis compared to the size of the staircase.

L shape: $\mathrm{LM}_\prec(\mathcal{G}) = \left\{x\,y, y^d, x^d\right\}$ in dimension 2 and $\mathrm{LM}_\prec(\mathcal{G}) = \left\{y\,z, x\,z, x\,y, z^d, y^d, x^d\right\}$ in dimension 3. This case is the worst for the number of sequence queries compared to the sizes of the staircase and the Gröbner basis.

Simplex: $\mathrm{LM}_\prec(\mathcal{G}) = \left\{y^d, x\,y^{d-1}, \ldots, x^d\right\}$ in dimension 2 and $\mathrm{LM}_\prec(\mathcal{G}) = \left\{z^d, y\,z^{d-1}, x\,z^{d-1}, \ldots, y^d, x\,y^{d-1}, \ldots, x^d\right\}$ in dimension 3, i.e. all the monomials of degree d. This case is the best for the number of sequence queries and the worst for the size of the Gröbner basis, both compared to the size of the staircase.

The polynomial variant of the BMS and Scalar-FGLM algorithms performs fewer arithmetic

[12] INRIA
[13] CryptoNext Security

operations than the others, for large d. More precisely, its number of operations appears to be linear in $(\#S)^2 = O(\#S(\#S + \#\mathcal{G}))$ in fixed dimension suggesting that the complexity estimate in Theorem 1.13 is pessimistic. This can be related to the uni-indexed case: the naive BM algorithm, based on naive extended Euclidean algorithm, is only quadratic in the number of sequence terms that one considers. This is the starting point of a work-in-progress of a new collaboration between Romain Lebreton[14] and me that I detail in Section 6.2.

[14] Université de Montpellier

Chapter 5

Quasi-commutative Gröbner bases

This chapter is dedicated to quasi-commutative polynomials, their link with *linear recurrence relations with polynomial coefficients*, or *P-relations* of a sequence and how to compute or guess them.

5.1 P-relations and quasi-commutative polynomials

Linear recurrence relations with polynomial, in the indices, coefficients is a large class of linear recurrence relations containing the class of C-relations. As a classical example, the binomial sequence $\left(\binom{i_1}{i_2}\right)_{(i_1,i_2)\in\mathbb{N}^2}$ satisfies, for all $(i_1,i_2)\in\mathbb{N}^2$,

$$(i_1-i_2+1)\binom{i_1+1}{i_2}=(i_1+1)\binom{i_1}{i_2},\quad (i_2+1)\binom{i_1}{i_2+1}=(i_1-i_2)\binom{i_1}{i_2}.$$

The Eval operator, see also Section 1.3, can be extended to deal with these relations by adding a new set of variables $\boldsymbol{\delta}=(\delta_1,\dots,\delta_n)$ with the property that $\delta_\ell = x_\ell\frac{\partial}{\partial x_\ell}$ for all $1\le\ell\le n$. The $2n$ variables satisfy the following quasi-commutative rules for all $1\le k,\ell\le n$, $k\neq\ell$,

$$\begin{aligned} x_kx_\ell &= x_\ell x_k, & \delta_k\delta_\ell &= \delta_\ell\delta_k,\\ x_k\delta_\ell &= \delta_\ell x_k, & \delta_k x_k &= x_k(\delta_k+1).\end{aligned}$$

As a consequence, the polynomials in $\boldsymbol{\delta}$ and $\boldsymbol{x}$ with coefficients in $\mathbb{K}$ form a quasi-commutative ring, denoted $\mathbb{K}[\boldsymbol{\delta}]\langle\boldsymbol{x}\rangle$.

These polynomials allow us to represent P-relations through the Eval operator

$$\operatorname{Eval}\left(\sum_{\boldsymbol{r}\in\mathcal{R},\boldsymbol{s}\in\mathcal{S}}\gamma_{\boldsymbol{r},\boldsymbol{s}}\boldsymbol{\delta}^{\boldsymbol{r}}\boldsymbol{x}^{\boldsymbol{s}},\boldsymbol{u}\right)=\sum_{\boldsymbol{r}\in\mathcal{R},\boldsymbol{s}\in\mathcal{S}}\gamma_{\boldsymbol{r},\boldsymbol{s}}\boldsymbol{s}^{\boldsymbol{r}}\boldsymbol{u}_{\boldsymbol{s}}.$$

Observe that $\boldsymbol{\delta}^{\boldsymbol{r}}\boldsymbol{x}^{\boldsymbol{s}}\boldsymbol{\delta}^{\boldsymbol{k}}\boldsymbol{x}^{\boldsymbol{i}}=(\boldsymbol{\delta}-\boldsymbol{s})^{\boldsymbol{k}}\boldsymbol{\delta}^{\boldsymbol{r}}\boldsymbol{x}^{\boldsymbol{s}+\boldsymbol{i}}$. Therefore, for any $g\in\mathbb{K}[\boldsymbol{\delta}]\langle\boldsymbol{x}\rangle$,

$$\operatorname{Eval}\left(g\boldsymbol{\delta}^{\boldsymbol{k}}\boldsymbol{x}^{\boldsymbol{i}},\boldsymbol{u}\right)=\boldsymbol{i}^{\boldsymbol{k}}\operatorname{Eval}\left(g\boldsymbol{x}^{\boldsymbol{i}},\boldsymbol{u}\right).$$

Thus, multiplying g on the right by $\boldsymbol{\delta}^{\boldsymbol{k}}\boldsymbol{x}^{\boldsymbol{i}}$ shifts the recurrence relation by $\boldsymbol{i}$ and multiplies it by $\boldsymbol{i}^{\boldsymbol{k}}$. We deduce that if g is such that $\operatorname{Eval}\left(g\boldsymbol{x}^{\boldsymbol{i}},\boldsymbol{u}\right)=0$ for all $\boldsymbol{i}\in\mathbb{N}^n$, then $\operatorname{Eval}\left(g\boldsymbol{\delta}^{\boldsymbol{k}}\boldsymbol{x}^{\boldsymbol{i}},\boldsymbol{u}\right)=0$ for all $\boldsymbol{k}\in\mathbb{N}^n$ and $\boldsymbol{i}\in\mathbb{N}^n$. Hence, the set of polynomials that represent P-relations satisfied by a sequence $\boldsymbol{u}$ is a right ideal of $\mathbb{K}[\boldsymbol{\delta}]\langle\boldsymbol{x}\rangle$ called the *ideal of P-relations of* $\boldsymbol{u}$ and denoted $\mathrm{I_P}(\boldsymbol{u})$.

From $\boldsymbol{\delta}^{\boldsymbol{r}}\boldsymbol{x}^{\boldsymbol{s}}\boldsymbol{\delta}^{\boldsymbol{k}}\boldsymbol{x}^{\boldsymbol{i}}=(\boldsymbol{\delta}-\boldsymbol{s})^{\boldsymbol{k}}\boldsymbol{\delta}^{\boldsymbol{r}}\boldsymbol{x}^{\boldsymbol{s}+\boldsymbol{i}}$, we also notice that for $\prec$, a monomial order on the variables $\boldsymbol{\delta}$ and $\boldsymbol{x}$, we have $\mathrm{LM}_\prec(gh)=\mathrm{LM}_\prec(g)\,\mathrm{LM}_\prec(h)$ for any two polynomials g and h. Furthermore, the notion of Gröbner basis can be defined for right ideals with the property that a finite subset $\mathcal{G}$ of an ideal I is a Gröbner basis I w.r.t. $\prec$ if for any $f\in I$, there exist $g\in\mathcal{G}$ and a monomial m such that $\mathrm{LM}_\prec(f)=\mathrm{LM}_\prec(g)m$.

In particular, Buchberger's algorithm and criteria can be extended to this setting, see [75].

Example 5.1: From $(i_1+1-i_2)\binom{i_1+1}{i_2}-(i_1+1)\binom{i_1}{i_2}=0$ and $(i_2+1)\binom{i_1}{i_2+1}-(i_1-i_2)\binom{i_1}{i_2}$, we deduce that $g_1=(\delta_1-\delta_2)x_1-(\delta_1+1)$ and $g_2=\delta_2x_2-(\delta_1-\delta_2)$ are in the ideal of P-relations of the binomial sequence.

Observe that in $\mathbb{K}(\boldsymbol{\delta})\langle\boldsymbol{x}\rangle=\mathbb{K}(\boldsymbol{\delta})\otimes\mathbb{K}[\boldsymbol{\delta}]\langle\boldsymbol{x}\rangle$, that is $\boldsymbol{\delta}$ and $\boldsymbol{x}$ still quasi-commute but we allow rational fractions in $\boldsymbol{\delta}$, these two polynomials are linear in $\boldsymbol{x}$ and linearly independant. They thus span a zero-dimensional ideal that contains $g_1x_2+g_2=(\delta_1-\delta_2)x_1x_2-(\delta_1-\delta_2+1)x_2-(\delta_1-\delta_2)=(x_1x_2-x_2-1)(\delta_1-\delta_2)$ and thus $x_1x_2-x_2-1$ representing the, so-called Pascal's rule, C-relation:

$$\forall\,(i_1,i_2)\in\mathbb{N}^2,\quad\binom{i_1+1}{i_2+1}=\binom{i_1}{i_2+1}+\binom{i_1}{i_2}.$$

5.2 Guessing P-relations

In the uni-indexed case and a sequence $\boldsymbol{u} = (u_i)_{i\in\mathbb{N}}$, one can compute a basis of $\mathrm{I_P}(\boldsymbol{u})$ using the Beckermann-Labahn algorithm [6] and its recent improvements [68], based on a divide-and-conquer approach. Many computer algebra systems propose an implementation of guessing P-relations algorithms, whether be it in the univariate or the multivariate case. We can cite for instance [99] in Maple [9], [71] in Mathematica [67]. Some also have dedicated arithmetic for quasi-commutative polynomials like Maple or SageMath [103] through a package [69].

5.2.1 A linear-algebra-based algorithm

With Jean-Charles Faugère[1,2], I designed an extension to the Scalar-FGLM algorithm in order to guess P-relations or C-relations between several sequences. For P-relations satisfied by $\boldsymbol{u}$, the idea is to build a generalized multi-Hankel matrix $H_{T,\Theta}$, where T and Θ are sets of monomials in respectively $\mathbb{K}[\boldsymbol{x}]$ and $\mathbb{K}[\boldsymbol{\delta}]\langle\boldsymbol{x}\rangle$. The entry at the intersection of the column labeled with $\boldsymbol{\delta}^{\boldsymbol{r}}\boldsymbol{x}^{\boldsymbol{s}}$ and the row labeled with $\boldsymbol{x}^{\boldsymbol{i}}$ is $\mathrm{Eval}\left(\boldsymbol{\delta}^{\boldsymbol{r}}\boldsymbol{x}^{\boldsymbol{s}}\boldsymbol{x}^{\boldsymbol{i}}, \boldsymbol{u}\right) = (\boldsymbol{s}+\boldsymbol{i})^{\boldsymbol{r}}u_{\boldsymbol{s}+\boldsymbol{i}}$. Therefore, a vector in the right kernel of this matrix represents a polynomial $G \in \mathbb{K}[\boldsymbol{\delta}]\langle\boldsymbol{x}\rangle$ such that $\mathrm{Eval}\left(G\boldsymbol{x}^{\boldsymbol{i}}, \boldsymbol{u}\right) = 0$ for all $\boldsymbol{x}^{\boldsymbol{i}} \in T$.

Example 5.2: Consider the binomial sequence $\left(\binom{i_1}{i_2}\right)_{(i_1,i_2)\in\mathbb{N}^2}$, $\Theta = \{1, \delta_2, \delta_1, x_2, x_1, \delta_2x_2, \delta_1x_2, \delta_2x_1, \delta_1x_1\}$ is the set of all monomials of bidegree at most $(1,1)$ in $\boldsymbol{\delta}$ and $\boldsymbol{x}$ for an order $\prec$ eliminating $\boldsymbol{x}$ and breaking ties on $\boldsymbol{\delta}$ and on $\boldsymbol{x}$ using $\prec_{\mathrm{DRL}}$. We take T to be the set of all monomials of degree at most 3 so that T is larger than Θ.

$$H_{T,\Theta} = \begin{array}{c} \\ 1 \\ x_2 \\ x_1 \\ x_2^2 \\ x_1x_2 \\ x_1^2 \\ x_2^3 \\ x_1x_2^2 \\ x_1^2x_2 \\ x_1^3 \end{array} \begin{array}{c} \begin{array}{ccccccccc} 1 & \delta_2 & \delta_1 & x_2 & x_1 & \delta_2x_2 & \delta_1x_2 & \delta_2x_1 & \delta_1x_1 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 2 & 0 & 0 & 2 & 4 \\ 1 & 0 & 2 & 2 & 1 & 2 & 4 & 0 & 3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 2 & 2 \\ 2 & 2 & 4 & 1 & 3 & 2 & 2 & 3 & 9 \\ 1 & 0 & 3 & 3 & 1 & 3 & 9 & 0 & 4 \end{pmatrix} \end{array}.$$

From this matrix, we can see the relations $\delta_2x_2 - (\delta_1 - \delta_2)$ and $(\delta_1 - \delta_2)x_1 - (\delta_1 + 1)$.

More rencently, a divide-and-conquer algorithm was designed and proposed in [82] which generalizes the Beckermann-Labahn algorithm to multi-indexed sequences.

5.2.2 An hybrid approach

In the combinatorics context, the ultimate goal is not always to guess the whole Gröbner basis of the ideal of P-relations for a specific monomial order, but rather to determine if the sequence is P-finite or not. For instance, a necessary condition is that the ideal of P-relations in $\mathbb{K}(\boldsymbol{\delta})\langle\boldsymbol{x}\rangle$ is zero-dimensional. Therefore, providing an adaptive algorithm that minimizes the number of sequence terms is of upmost importance.

In a similar fashion as in Section 4.2.3, we can discover, step by step, which monomials are in the staircase of the ideal of P-relations of $\mathbb{K}[\boldsymbol{\delta}]\langle\boldsymbol{x}\rangle$ of the input sequence for the input monomial order $\prec$. In [16], Jean-Charles Faugère[1,2] and I proposed an hybrid approach based on Gröbner bases computations for quasi-commutative polynomials to discover new P-relations without any extra queries to the sequence. The idea is that if two polynomials $g_1, g_2 \in \mathbb{K}[\boldsymbol{\delta}]\langle\boldsymbol{x}\rangle$ encode P-relations satisfied by the sequence, then any polynomial in $\langle g_1, g_2\rangle$ also encodes a P-relation. Therefore, as soon as two P-relations g_1 and g_2 are guessed, the goal is to compute a Gröbner basis $\{g_1, g_2, \ldots, g_r\}$ of $\langle g_1, g_2\rangle$. This will yield polynomials, namely $g_3, \ldots, g_r$, whose leading monomials are not in $\langle \mathrm{LM}_\prec(g_1), \mathrm{LM}_\prec(g_2)\rangle$. The advantage of this method is twofold. First, since $\mathrm{LM}_\prec(g_3), \ldots, \mathrm{LM}_\prec(g_r) \succ$

[1] INRIA
[2] CryptoNext Security

$\mathrm{LM}_\prec(g_1), \mathrm{LM}_\prec(g_2)$, they require more queries to the sequence to be correctly guessed. Yet, such a Gröbner basis computation does not require any more queries. Then, these P-relations may help us determine that the ideal of P-relations is 0-dimensional in $\mathbb{K}(\boldsymbol{\delta})\langle \boldsymbol{x}\rangle$, which is a necessary condition for the table to be P-finite.

5.3 Structured sequences

In many applications, like in enumerative combinatorics with 2D/3D-walks, the studied sequences are highly structured. For instance, they have a lot of zero-terms. From a linear algebra viewpoint, these zero terms induce void conditions on the C-relations or P-relations and thus drops of ranks in the generalized multi-Hankel matrix that one deals with to guess the relations. Therefore, we need to build matrices with many more rows than columns to recover *correct* relations, i.e. C-relations or P-relations that do not prove to be fake after further testings.

The goal is thus to only deal with the nonzero terms from the beginning. These nonzero terms lie in general in a cone $\mathscr{C}$ which is a *submonoid* of $\mathbb{N}^n$, i.e. $0 \in \mathscr{C}$ and for all $\boldsymbol{i}, \boldsymbol{j} \in \mathscr{C}$, $(\boldsymbol{i}+\boldsymbol{j}) \in \mathscr{C}$.

In [21], Mohab Safey El Din[3] and I extended the Scalar-FGLM algorithm, and its variants, adaptive and for P-relations, so that it only considers terms of a sequence lying in a cone. In this paper, we make the connexion with sparse Gröbner bases as defined in [7, 56] and extend them to the context of quasi-commutative polynomials in $\mathbb{K}[\boldsymbol{\delta}]\langle \boldsymbol{x}\rangle$ for P-relations.

Given such a cone $\mathscr{C}$ and polynomials with support in its associated set of monomials $\mathscr{T}(\mathscr{C}) = \left\{\boldsymbol{x}^{\boldsymbol{i}} \in \mathscr{T} \middle| \boldsymbol{i} \in \mathscr{C}\right\}$, one may want to perform all the polynomial operations with monomials in $\mathscr{T}(\mathscr{C})$ in order to take advantage of the structure of the support when computing a Gröbner basis of the ideal they span. While this is not always possible, one can achieve this goal by considering the ideal the polynomials span in the algebra

$$\mathbb{K}[\mathscr{C}] = \left\{ f = \sum_{s\in\mathscr{C}} f_s \boldsymbol{x}^s \;\middle|\; \operatorname{supp} f \text{ is finite} \right\}.$$

The ideal spanned by $f_1, \ldots, f_m \in \mathbb{K}[\mathscr{C}]$ is defined as

$$\langle f_1, \ldots, f_m\rangle_{\mathscr{C}} = \left\{ \sum_{k=1}^{m} f_k q_k \;\middle|\; q_1, \ldots, q_m \in \mathbb{K}[\mathscr{C}] \right\}.$$

Given a monomial order $\prec$, a $\prec$-sparse Gröbner basis of $\langle f_1, \ldots, f_m\rangle_{\mathscr{C}}$ is a finite subset thereof whose properties extend those satisfied by a Gröbner basis, in the context of the algebra $\mathbb{K}[\mathscr{C}]$. In particular, the divisibility property on the leading monomials in $\mathscr{T}(\mathscr{C})$ is still satisfied.

Definition 5.3 ([56, Def. 3.1] and [7, Def. 3.3])**:** Let $\mathscr{C} \subseteq \mathbb{N}^n$ be a submonoid, $f_1, \ldots, f_m \subseteq \mathbb{K}[\mathscr{C}]$ be polynomials and $\prec$ be a monomial order. Let $I = \langle f_1, \ldots, f_m\rangle_{\mathscr{C}}$. Then, a *$\prec$-sparse Gröbner basis* of I is a generating set $\mathscr{G} = \{g_1, \ldots, g_t\} \subseteq \mathbb{K}[\mathscr{C}]$ such that for all $f \in I$, $\mathrm{LM}_\prec(f) = \mathrm{LM}_\prec(g)\boldsymbol{x}^{\boldsymbol{i}}$ for some $g \in \mathscr{G}$ and $\boldsymbol{x}^{\boldsymbol{i}} \in \mathscr{C}$.

The associated staircase of $\mathscr{G}$ is the set of monomials s in $\mathscr{T}(\mathscr{C})$ such that for any $g \in \mathscr{G}$, there is no monomial $\boldsymbol{x}^{\boldsymbol{i}} \in \mathscr{T}(\mathscr{C})$ such that $s = \mathrm{LM}_\prec(g)\boldsymbol{x}^{\boldsymbol{i}}$.

Let us notice that for $\mathscr{C} = \mathbb{N}^n$, $\mathbb{K}[\mathscr{C}] = \mathbb{K}[\mathbb{N}^n] = \mathbb{K}[\boldsymbol{x}]$ and sparse Gröbner bases are classical Gröbner bases. Furthermore, like classical Gröbner bases, sparse Gröbner bases allow one to solve the ideal membership problem in $\mathbb{K}[\mathscr{C}]$ in an effective way.

Example 5.4: The cone $\mathscr{C} = \left\{(i_1, i_2) \in \mathbb{N}^2 \middle| i_1 \le 2i_2, i_2 \le 2i_1\right\}$ is spanned by $(1,1)$,$(2,1)$ and $(1,2)$ as a submonoid of $\mathbb{N}^2$. Although x_1x_2 divides both $x_1^2x_2$ and $x_1x_2^2$ in $\mathbb{K}[\boldsymbol{x}]$, it does not in $\mathbb{K}[\mathscr{C}]$ as the respective quotients are x_1 and x_2, which are not in $\mathscr{T}(\mathscr{C})$.

Theorem 5.5 (see also [21, Th. 3.2])**:** Let $\mathscr{C}$ be a submonoid cone of $\mathbb{N}^n$ spanned by a finite minimal set of generators. Let $\prec$ be a monomial ordering on $\mathscr{T}$, the set of monomials in n variables, and let $T \subset \mathscr{T}(\mathscr{C})$ be a set of monomials ordered for $\prec$ stable by division.

Then, the Scalar-FGLM algorithm called on sequence $\boldsymbol{u}$, T and $\prec$ returns a set of polynomials G with support in $\mathscr{T}(\mathscr{C})$, such that for all $s \in T \setminus \langle \mathrm{LM}_\prec(G)\rangle$, s is in the associated staircase of a

[3] Sorbonne Université

$\prec$-sparse Gröbner basis of $\mathrm{I}_C(\boldsymbol{u})$.

Furthermore, if the ideal of C-relations of $\boldsymbol{u}$ is 0-dimensional and has a reduced $\prec$-sparse Gröbner basis with support in T, then the output of the Scalar-FGLM algorithm called on $\boldsymbol{u}$ and T is this reduced $\prec$-sparse Gröbner basis.

As an illustration, we consider walks in 2D- and 3D-spaces, namely Gessel planar walk $\boldsymbol{g}$ in the nonnegative quadrant $\mathbb{N}^2$ with steps in $\{(1,0),(1,1),(-1,0),(-1,-1)\}$ and the 3D-space Walk-43 $\boldsymbol{w}$ of [25] in the nonnegative octant $\mathbb{N}^3$ with steps in $\{(-1,-1,-1),(-1,-1,1),(-1,1,0),(1,0,0)\}$. In particular, we restrict ourselves to a subsequence of each where one index is 0. These walks come naturally with a cone structure: for instance whenever $n \neq 2n' + 2j$, then $g_{n,0,j} = 0$. Likewise, whenever $n \neq 8n' + 2j + 4k$, then $w_{n,0,j,k} = 0$. Thus, it makes sense to look for the relations given by the sequence terms $g_{2n'+2j,0,j}$ and $w_{8n'+2j+4k,0,j,k}$.

Type	Cone				Full Orthant			
	Matrix size	Queries	Relations		Matrix size	Queries	Relations	
			Fake	Correct			Fake	Correct
$g_{n,0,j}$	444×441	866	11	0	496×495	946	48	0
$g_{n,0,j}$	631×564	1 174	0	0	$1\,326 \times 661$	1 942	84	0
$g_{n,0,j}$	721×711	1 408	15	8	726×715	1 386	67	0
$g_{n,0,j}$	$1\,951 \times 1\,089$	3 010	0	21	$2\,556 \times 1\,001$	3 491	136	6
$w_{n,0,j,k}$	223×211	430	7	1	220×210	395	24	0
$w_{n,0,j,k}$	444×253	552	2	1	680×267	912	37	0
$w_{n,0,j,k}$	406×400	799	11	6	406×400	771	27	0
$w_{n,0,j,k}$	806×522	1 320	2	6	$1\,540 \times 589$	2 073	68	0

Table 5.1: Guessing fake and correct P-relations

In Table 5.1, we can see that considering only terms in the cone allows us to almost only discover correct relations compared to the full orthant case, even when the generalized multi-Hankel matrices are almost square.

Chapter 6

Research project

I want to develop my research project on computing and guessing Gröbner bases in the commutative and quasi-commutative setting relying on MSOLVE as a validation tool. The main goal is to develop general implementations on the one hand and specific ones for applications from combinatorics, cryptography or robotics on the other hand.

I present my research project in three axes. The first one, in Section 6.1, is about MSOLVE and its development. I see it as a transverse axis. The goal is to implement existing algorithms presented in the previous chapters but also new ones that shall be designed in the other axes. Furthermore, I shall exploit high-performance computing hardware to enhance the computation capabilities of MSOLVE.

As a second axis, Section 6.2, I aim to accelerate guessing algorithms, especially for C-relations, in order to provide theoretical and practical complexity estimates on the SPARSE-FGLM algorithms when roots have multiplicities. In particular, we will trade the information on the multiplicities for the efficiency and vice versa, when a second multiplication matrix is required. A polynomial matrix extension is also targeted.

The last axis, Section 6.3, is dedicated to Gröbner bases for quasi-commutative polynomials in order to deal with P-relations. The main objective is to bring the complexity of $\prec_{\text{DRL}}$-Gröbner bases for quasi-commutative polynomials to that of classical polynomials.

6.1 MSOLVE and fundamental algorithms

A first goal for MSOLVE is to increase its functionalities by implementing state-of-the-art techniques for general ideals which are not covered yet. Furthermore, at the moment the modular implementation uses native integer types which allow us to handle large characteristic but with computations that are slower than those using floating-point types.

6.1.1 Types for the coefficients

For the moment, MSOLVE uses native integer types to represent modular integers with primes of size at most 31 bits. While some fast Central Processing Unit (CPU) instructions (like AVX2 or AVX512) are available for integer types, current CPUs are such that even faster instructions are available to floating-point types. These types allow us to handle modular arithmetic for lesser prime (up to 26 bits for double-precision floating-point numbers). Assuming not too many bad primes are picked during a multi-modular Gröbner basis computation, using these types could speed the computations up before lifting the result over $\mathbb{Q}$.

Likewise, for cryptographic applications, we shall need a dedicated implementation over $\mathbb{F}_2$. In fact, we may have to introduce ways to compute over non-prime finite fields. This is why, I want to integrate fast univariate polynomial-matrix operations in order to deal with computations over finite extension of finite fields, defined by a single polynomial.

6.1.2 Gröbner bases for a degree order

While the classical framework of polynomial system solving relies on computing first a $\prec_{\text{DRL}}$-Gröbner basis, see Figure 1.1, in some applications, a monomial order $\prec$ distinct from $\prec_{\text{DRL}}$ might be more convenient for the computation. This can be a 2-block $\prec_{\text{DRL}}$-order, which is already implemented in MSOLVE, or a weighted degree order, see also [54, 55, 105]. When the input system is quasi-homogeneous for a system of weights, one can take advantage of this quasi-homogeneity to improve the runtime and the complexity estimates of the F_4 and F_5 algorithms [47, 48] by a factor depending on the products of the weight. We want to benefit from this by implementing this kind of weighted degree monomial orders in MSOLVE in order to tackle systems coming from applications such as robotics where the weights are naturally coming from dimensional homogeneity.

Another approach to speed Gröbner bases computations up for (quasi-)homogeneous systems is to use *Hilbert-driven* Gröbner bases algorithms. These algorithms, such as Traverso's [104], see

also [41, Chap. 10, Sec. 2], assume that the Hilbert series of the ideal is already known and provide a criterion based on the degree of a critical pair to discard it without computing its reduction to 0. Combined with the F_4 algorithm, this allows us to handle Macaulay-like matrices with both fewer rows and fewer columns. Note, however, that the expected speed-ups may only be observed in positive characteristic or, in the case of characteristic 0, during the learning phase modulo the first prime.

6.1.3 Change of order

For the moment, the implemented change of order step relies on the sparse-FGLM algorithm [52, 53] in the shape position case. This assumption rises issues when the solutions of the system have multiplicities: even after introducing a generic linear form, the ideal might not be in shape position. Thus we may not compute the $\prec_{\text{LEX}}$-Gröbner basis of the ideal. While we can compute the parametrizations of the solutions, through the radical of the ideal, this makes us lose the information on the multiplicity. To this end, we want to provide more general change of order algorithms in MSOLVE. A first step will be to implement the seminal FGLM algorithm [50] before considering the algorithms I developed based on linear algebra and in particular, the adaptive algorithm of [11, 12], see also Section 4.2.2.

In order to provide an efficient implementation of the guessing algorithms, I want to use the structure of the multi-Hankel matrix and in particular, its *quasi-Hankel* structure. Indeed, if the univariate polynomial in x_n in the reduced $\prec_{\text{LEX}}$-Gröbner basis has a large degree, i.e. of the same magnitude order as the degree of the ideal D, the multi-Hankel matrix will be quasi-Hankel with a small displacement rank. We can thus compute its vector in its kernel, and thus C-relations, fast using [26, 27]. Furthermore, depending on the geometry of the $\prec_{\text{DRL}}$- and $\prec_{\text{LEX}}$-staircases, computing the coefficients of this multi-Hankel matrix can just require Algorithm 2.2 with only one matrix, the one of the multiplication by x_n, but extra column-vectors. All-in-all, when D is exponential in n, the expected complexity will still be dominated by the computation of the vectors of Algorithm 2.2 in $O(tD^2)$ operations.

In addition to this, I want to implement a multi-modular approach for sequences with coefficients in $\mathbb{Q}$. The idea is that the purpose of the adaptive algorithm is to find, at a low cost, the support of the reduced $\prec_{\text{LEX}}$-Gröbner basis. Once this has been computed modulo a first prime, we can switch to the general SCALAR-FGLM with one big multi-Hankel matrix whose rows and columns are labeled by the monomials in this support. This shall allow us to compute exactly the number of sequence terms required to recover the sought Gröbner basis modulo subsequent primes.

6.1.4 Saturation and colon ideals

With my coauthors, we want to push forward the efficiency of F_4SAT by investigating computational tricks to avoid "computing zero". For instance, we want to provide an early termination criterion to reduce the time spent in the last saturation step.

Likewise, we want to study a signature approach, as in Faugère's F_5 algorithm [44, 48], to minimize the reductions to 0 of S-polynomials.

I also want to provide a multi-modular approach and a tracer for SPARSE-FGLM-COLON for systems over $\mathbb{Q}$ in MSOLVE. The goal is to learn how to minimize Σ and Σ' to the actual set of required monomials to write the $\prec_{\text{LEX}}$-Gröbner basis of the colon ideal but also to compute the exact number of necessary sequence terms in the Wiedemann part of the algorithm. This will make computations in the apply phase modulo all the primes but the first one optimal and thus the fastest implementation.

6.1.5 Exact solving on a GPU

Graphics Processing Units (GPUs) are, by design, well-suited to process large blocks of data in parallel and thus to perform linear algebra routines, more so than CPUs. Furthermore, they natively handle double-precision floating-point number arithmetic but only simulate long integer ones through short integer arithmetic, which comes with an overhead. For instance, NVIDIA CUDA

and TENSOR cores, natively, only have 8-bit integer types, whereas they, natively, have 64-bit floating-point types[1]. Furthermore, it has been shown that GPUs are very efficient for correctly rounding functions evaluations [59], reinforcing even more the advantage of using floating-point arithmetic in order to simulate a modular one. Thus, we aim to take advantage of their computational power to transpose the linear algebra code of our polynomial system solver MSOLVE to efficiently work with GPUs.

The main objective of this task is the design of fast Gröbner bases computation algorithms, based on high performance linear algebra algorithms, in particular exploiting GPUs, and their integration into MSOLVE in order to tackle applications challenges such as multivariate cryptography or robotics. This is Dimitri Lesnoff[2]'s Ph.D. subject, who I have been supervising jointly with Stef Graillat[3] and Théo Mary[4] since October 2022. We shall first revisit modular arithmetic at the core of exact algorithms relying on fast low-precision arithmetics such as `fp16`, `bfloat16`, `fp8`, ...for floating-points numbers and `int8`, `int4` for integers. This part will be done in parallel for CPUs, as in Section 6.1.1 and for GPUs. Then, we will adapt the block-Wiedemann approach [40, 66] of the SPARSE-FGLM algorithm. Since the matrix at hand is very particular, it can be seen as the concatenation of a permutation matrix and a dense matrix after reordering the columns, we will study how to balance more efficiently the CPU load and the GPU load to iterate the product of this matrix with some vectors or very thin matrices. We also target the design of a sparse matrix arithmetic which is both efficient for handling Macaulay matrices for the F_4 algorithm and dedicated to a GPU or a CPU + GPU architecture.

In order to tackle very large problems whose solving will unlock new advances in critical applications, we have to handle large matrices that cannot even be stored on the GPU. We plan to devise new algorithms that exploit a memory representation of these Macaulay matrices that suits our computations but also the limited RAM, a few tens of gigabytes, of a graphic card. Moreover, computing Gröbner bases at scale will require the use of multiple GPUs and CPUs in parallel. We will therefore work on making the algorithms scalable in a parallel context. Notably, we will minimize memory communication between the CPU and the GPU, by adapting cache-oblivious storing and algorithms [2] to a larger scale such as the RAM of the graphic card.

6.2 Faster change of order and guessing C-relations

6.2.1 Guessing faster

While the polynomial point of view allows us to bring the guessing of C-relations for multi-indexed sequences closer to the fast guessing for uni-indexed sequences, there is still a complexity gap. Indeed, the stated complexity in Theorem 1.13 becomes cubic, instead of quasi-linear, for uni-indexed sequences. This is due to the complexity analysis that relies, first, on a linear number of polynomial reductions and, second, on naive polynomial reductions. In fact, this cubic complexity stated in Theorem 1.13 comes from an overestimation on the number of needed polynomials to reduce a new one, while in practice, this is not observed, especially on sequences of prescribed ideal of C-relations but with *random* initial terms. Thus, as an on-going collaboration with Romain Lebreton[5], I want to improve this algorithm by handling univariate polynomials instead of multivariate ones, in order to take advantage or their fast arithmetic while reestimating the number of reductions to target a complexity which is only quadratic in the number of input sequence terms or the size of the output Gröbner basis.

As a second goal, I want to guess P-relations using polynomial arithmetic instead of linear algebra [16]. From the generating series of a sequence and its derivatives, or more precisely the mirror polynomials of a truncation of these series, the goal is to find algebraic combinations thereof which are *small* modulo the monomial ideal $\langle x_1^{D_1}, \ldots, x_n^{D_n} \rangle$, where $D_1, \ldots, D_n$ depend on the sequence terms we allow ourselves to use.

[1] https://www.nvidia.com/en-us/data-center/tensor-cores/
[2] Ph.D. student at Sorbonne Université
[3] Sorbonne Université
[4] CNRS
[5] Université de Montpellier

Efficiency of guessing algorithms is based on two aspects: the number of performed operations and the number of sequence terms that are needed. Indeed, in many applications, computing the sequence is the bottleneck. Thus, to make this approach the most efficient, we shall closely look at the number of different sequences terms that are needed to correctly guess the relations. Furthermore, to optimize the number of operations, we shall rely on efficient algorithms for univariate polynomials and uni-indexed sequences. The main goal is to reach a complexity at most quadratic in the size of the output instead of only cubic.

These guessing algorithms may find fake relations, this happens in general when too few terms are used or when most of the terms are 0. This can be circumvent by *structured* guessing relations using mainly the nonzero sequence terms as in [21]. We will pay attention to these bad sequences so that our new algorithm avoids these fake relations as much as possible.

6.2.2 Guessing radical ideals

Radical ideals are important in the polynomial system solving applications as the solutions have all multiplicities 1. Therefore, numerical methods to approximate the real or complex solutions behave much better with this kind of ideals.

In collaboration with Alin Bostan[6], Manuel Kauers[7] and Christoph Koutschan[8], I want to investigate an extension of [29] to more general radical ideals. Indeed, in this paper, given a sequence $\boldsymbol{u}$, the authors propose an algorithm for guessing the $\prec_{\text{LEX}}$-Gröbner basis of $\sqrt{\text{I}_{\text{C}}(\boldsymbol{u})}$ if it is in shape position. Yet, the output is meaningless if this radical is not in shape position. While BMS and Scalar-FGLM allow us to guess $\text{I}_{\text{C}}(\boldsymbol{u})$, this may require many sequence terms, especially if the ideal has a large degree. Thus, we want to propose a trade-off between the number of queries to the sequence and the quality of the output by allowing to guess polynomials only in $\sqrt{\text{I}_{\text{C}}(\boldsymbol{u})}$.

6.2.3 Guessing with multiplicities

In some situations where the system has roots with multiplicities, the sought $\prec_{\text{LEX}}$-Gröbner basis *cannot* be in shape position, even after a generic linear change of variables. Such a system is called 2-*thick* in [5] and it requires a second structured $D\times D$-matrix, of a similar kind, to be computed: the matrix of the multiplication by x_{n-1}, which has $O(\tau D)$ nonzero coefficients. Furthermore, in most situations the stability property is still satisfied which means that the computation of this second matrix is cheap. However, it does not ensure (actually it *almost never can*) that this second matrix is computed for free.

A first goal is to derive a sharp complexity estimate on the computation of this second matrix based on the $\prec_{\text{DRL}}$-Gröbner basis, exploiting the stability property and the work of Moreno-Socías [80], for the first matrix. For instance, whenever $\left\langle \text{LM}_{\prec_{\text{LEX}}}(I)\right\rangle = \left\langle x_n^{D_n}, x_{n-1}^{D_{n-1}}, x_{n-2}, \dots, x_1\right\rangle$ or $\left\langle \text{LM}_{\prec_{\text{LEX}}}(I)\right\rangle = \left\langle x_n^{D_n}, x_n x_{n-1}, x_{n-1}^2, x_{n-2}, \dots, x_1\right\rangle$ which seem to be simplest cases.

As a by-product, we will obtain complexity bounds on the computation of the sequence terms that appear in the multi-Hankel matrix built by Scalar-FGLM [11, 12] to recover the $\prec_{\text{LEX}}$-Gröbner basis. Then, as a second goal, we will rely on the quasi-Hankel structure of this multi-Hankel matrix, and fast algorithms for quasi-Hankel matrices [26, 27], to analyze the complexity on the computation of the sought $\prec_{\text{LEX}}$-Gröbner basis. All in all, we will have a complete description and complexity estimate of Faugère and Mou's [52, 53] Sparse-FGLM algorithm for generic 2-thick systems.

6.2.4 Polynomial matrices with multiplicities

A first goal is to adapt the polynomial-matrix algorithm for the change of order, see [20] and Section 3.1.2, to take into account structures. For instance, in [57] and [21], the authors tweak respectively the FGLM and the Sparse-FGLM algorithms for ideals that are globally invariant under the action of a finite abelian group. The main idea is to split the multiplication matrices into block-matrices, improving the complexity by a factor that depends on the size of the group.

Another objective is to be able to compute the $\prec_{\text{LEX}}$-Gröbner basis of a zero-dimensional ideal

[6] INRIA
[7] Johannes Kepler Universität Linz
[8] Österreichische Akademie der Wissenschaften

$I : \langle \varphi \rangle$ from the $\prec_{\text{DRL}}$-Gröbner basis of a, potentially positive-dimensional, ideal I using also the polynomial-matrix algorithm for change of order. We also want to investigate the computation of a parametrization of the radical ideal when I is not radical, in order to remove multiplicities. How can we take advantage of the polynomials to add to I to span $\sqrt{I}$ in this polynomial matrix in order to compute the $\prec_{\text{LEX}}$-Gröbner basis of $\sqrt{I}$.

Then, a last objective, as a follow-up to Section 6.2.3, is to deal with the matrix of the multiplication by x_{n-1} in this polynomial matrix change of order algorithm. That is, we will develop this approach by considering a polynomial matrix $P_{x_{n-1}}$ instead of $M_{x_{n-1}}$. However, this matrix should lie in $\mathbb{K}[x_{n-1}, x_n]$ and thus, efficient bivariate-polynomial matrices operations would have to be developed in order to make this algorithm faster, theoretically and practically, than the SPARSE-FGLM algorithm.

6.2.5 Applications to guessing

I also want to target applications such as coding theory and optimization problems. In the former case, I want to extend the existing algorithms for guessing recurrences in order to decode extensions of cyclic codes such as those defined on an algebraic variety. As a first step, I will consider algorithms relying on linear algebra in order to understand the structure of the matrix at hand. In the latter case, I want to adapt our algorithms, in particular SCALAR-FGLM, to handle numerical sequences coming from the moment approach [64]. The main difficulty will be to return non-trivial relations and we shall take inspiration from approximate GCD algorithms to overcome it, see [22].

6.3 Algorithms for quasi-commutative Gröbner bases

6.3.1 Moment approach

As a starting collaboration with Lorenzo Baldi[9] and Pierre Lairez[10], we want to use the theory of holonomic functions, or sequences satisfying P-relations, to provide a general and efficient method for computing the moments of measures that are useful for the applications. To do so, we want to exploit the fact that when computing the volume defined by several polynomial inequalities $f_1, \dots, f_r \geq 0$ in the hypercube $[0,1]^n$, we are led to computing the integrals $\int_{[0,1]^n} f_1^{k_1} \cdots f_r^{k_r} \mathrm{d}\boldsymbol{x}$ for many indices $k_1, \dots, k_r$. Yet, the sequence of integrals satisfies P-relations. These relations are fundamental and we want to guess them by computing first a few moments and then exploit them to compute more moments in order to find larger relations.

6.3.2 Efficient Gröbner bases computation

Another topic I want to investigate is the computation of Gröbner bases for a total degree monomial order of ideals in a quasi-commutative setting and its theoretical and practical efficiency in relation with Section 5.2.2. A longer term goal is then to implement Gröbner bases of quasi-commutative polynomials into MSOLVE in order to solve applications from combinatorics and physics.

Following [75] and the generalization of Buchberger's criteria, the goal will be to dive into the understanding of how Faugère's F_4 algorithm [47] behaves or can be extended from the commutative setting to this one.

To do so, I want to study the module of trivial syzygies in order to get information on the sizes of the matrices that are built in F_4. What kind of information the commutation rules provide on the syzygies?

Hilbert series and Hilbert polynomials are powerful tools that allow one to understand the complexity of computing Gröbner bases. In the commutative case, one can derive a bound on the degree of the polynomials in a reduced Gröbner basis for a total degree order, thanks to them, see [79, Section 4.5, Corollary]. We will investigate how knowing in advance the Hilbert series can speed the Gröbner bases computations up, or how together with the Hilbert polynomials, they can give us a bound on the degrees of the polynomials in the reduced $\prec_{\text{DRL}}$-Gröbner basis.

[9] Post-doc at Sorbonne Université
[10] INRIA

In the particle physics and algebraic statistics application, computing representants of the quotient $\mathbb{K}(\delta)\langle x\rangle / J$, where J is a 0-dimensional ideal allows one to determine the twisted cohomology defined by a likelihood function, see [78]. In this paper, the authors need to compute the *contiguity matrices* of J, which actually correspond to the matrices of multiplication by $x_1, \ldots, x_n$ in $\mathbb{K}(\delta)\langle x\rangle / J$. Therefore, I want to study how to efficiently build these matrices from a $\prec_{\text{DRL}}$-Gröbner basis of J. Furthermore, depending on the shape of the $\prec_{\text{DRL}}$-staircase, some of these matrices are obtained from free, for instance the one for x_n, like in the commutative case, see [80]. As a longer term goal, I would like to study these conditions in this setting.

Appendix A

Ideals of C-relations of small degrees

A.1 Uni-indexed sequences

For $d \in \mathbb{N}^*$, it is clear that $\mathrm{S_C}\left(\langle x^d \rangle\right) = \{(\alpha_0, \ldots, \alpha_{d-1}, 0, 0, \ldots) \mid \alpha_0, \ldots, \alpha_{d-1} \in \mathbb{K}\}$, whereas for $a \in \mathbb{K}^*$, it is well-known that $\mathrm{S_C}\left(\langle (x-a)^d \rangle\right) = \left\{\left(\left(\alpha_0 + \cdots + \alpha_{d-1} i^{d-1}\right) a^i\right)_{i \in \mathbb{N}} \middle| \alpha_0, \ldots, \alpha_{d-1} \in \mathbb{K}\right\}$. Theses sequences $\left(\alpha(i)a^i\right)_{i\in\mathbb{N}}$ generalize the class of *geometric sequences* with initial term α_0 and common ratio a, which are the case $d = 1$, and the class of *arithmetic sequences* with initial term α_0 and common difference α_1, which are the case $d = 2$ and $a = 1$. For the sake of the completeness, we recall that if $\boldsymbol{u} \in \mathrm{S_C}\left(\langle (x-a)^d \rangle\right)$ for $a \in \mathbb{K}$ and $\boldsymbol{u} \notin \mathrm{S_C}\left(\langle (x-a)^{d-1} \rangle\right)$, then $\mathrm{I_C}(\boldsymbol{u}) = \langle (x-a)^d \rangle$. Such a sequence will be denoted $\mathfrak{u}_{a,d}$.

Using Theorem 1.5, we can deduce the general case.

Theorem A.1: Let $\mathbb{K}$ be algebraically closed and $g \in \mathbb{K}[x]$. Assume g factors as

$$g = \prod_{k=1}^{m} (x - a_k)^{d_k},$$

then $\boldsymbol{u} \in \mathrm{S_C}(\langle g \rangle)$ if, and only if, there exist unique $\mathfrak{u}_{a_1,e_1}, \ldots, \mathfrak{u}_{a_m,e_m}$ with $e_1 \leq d_1, \ldots, e_m \leq d_m$ such that

$$\boldsymbol{u} = \sum_{k=1}^{m} \mathfrak{u}_{a_k,d_k}.$$

As we can see, the roots of the polynomial, and their multiplicities, completely determine the form of the sequence terms.

A.2 Bi-indexed sequences

From the uni-indexed case and small computations, we can see that $\mathrm{S_C}\left(\langle (x_1 - a_1)^{d_1}, (x_2 - a_2)^{d_2} \rangle\right)$ is the vector space of sequences $\mathfrak{u}_{a_1,e_1} \otimes \mathfrak{u}_{a_2,e_2} = \left(\alpha_1(i_1)\alpha_2(i_2) a_1^{i_1} a_2^{i_2}\right)_{(i_1,i_2)\in\mathbb{N}^2}$ where $e_1 \leq d_1$, $e_2 \leq d_2$ and $\deg \alpha_1 = e_1$, $\deg \alpha_2 = e_2$. While such a polynomial ideal has (a_1, a_2) as its unique root with multiplicity $d_1 d_2$, this does not encompass all the cases. Before diving into multiplicities, we shall deal with the case of distinct single roots.

To simplify the presentation, we mostly consider the case $a_1 \neq 0$ and $a_2 \neq 0$, but the results on the ideals still hold whenever a_1, a_2 or both are 0.

A.2.1 Single roots

Let us consider two distinct points $(a_1, a_2), (b_1, b_2) \in \mathbb{K}^2$, then we can look at the ideal of polynomials vanishing on these points. This is the ideal $I = \langle (x_2 - a_2)(x_2 - b_2), (a_2 - b_2)(x_1 - a_1) - (a_1 - b_1)(x_2 - a_2), (x_1 - a_1)(x_1 - b_1) \rangle$. Observe how, if $a_2 \neq b_2$, the first two polynomials form a Gröbner basis of I for $x_2 \prec x_1$, otherwise the last two do. Hence, $\mathrm{S_C}(I)$ is a two-dimensional vector space and $\boldsymbol{u} \in \mathrm{S_C}(I)$ is uniquely determined by $u_{0,0}$ and, $u_{0,1}$ in the former case or $u_{1,0}$ in the latter one.

More generally, the sequence obtained as a linear combination, with nonzero coefficients, of the sequences $(a_{1,1}^{i_1} a_{1,2}^{i_2})_{(i_1,i_2)\in\mathbb{N}^2}, \ldots, (a_{k,1}^{i_1} a_{k,2}^{i_2})_{(i_1,i_2)\in\mathbb{N}^2}$ has its ideal of C-relations which is exactly the radical ideal of all polynomials vanishing on $(a_{1,1}, a_{1,2}), \ldots, (a_{k,1}, a_{k,2})$.

It remains to deal with multiplicities. As we shall see, even the multiplicity-2 case is broader than for uni-indexed sequences.

A.2.2 Root of multiplicity 2

The sequences whose ideal of C-relations only admits (a_1, a_2) as a root, and with multiplicity 2, are all of the type $\boldsymbol{u} = \left((\alpha_{0,0} + \alpha_{1,0} i_1 + \alpha_{0,1} i_2) a_1^{i_1} a_2^{i_2}\right)_{(i_1,i_2)\in\mathbb{N}^2}$ with $\alpha_{1,0}$ and $\alpha_{0,1}$ not both 0. The ideal of C-relations of such a sequence $\boldsymbol{u}$ is

$$J_{(a_1,a_2),2,(\alpha_{1,0},\alpha_{0,1})} = \left\langle (x_1 - a_1)^2, \alpha_{0,1}\hat{a}_2(x_1 - a_1) - \alpha_{1,0}\hat{a}_1(x_2 - a_2), (x_2 - a_2)^2 \right\rangle,$$

where $\hat{b}$ equals 1 if $b = 0$ and b otherwise. Notice that if $\alpha_{1,0} \neq 0$, the first two polynomials span the ideal, otherwise $\alpha_{0,1} \neq 0$ and the last two polynomials span it.

A.2.3 Root of multiplicity 3

The sequences whose ideal of C-relations only admits (a_1, a_2) as a root, and with multiplicity 3, are all of the type $\boldsymbol{u} = \left((\alpha_{0,0} + \alpha_{1,0} i_1 + \alpha_{0,1} i_2 + \alpha_{2,0} i_1^2 + \alpha_{1,1} i_1 i_2 + \alpha_{0,2} i_2^2) a_1^{i_1} a_2^{i_2}\right)_{(i_1,i_2)\in\mathbb{N}^2}$ with $\alpha_{2,0}\alpha_{0,2} = 4\alpha_{1,1}^2$ and $\alpha_{2,0}$ and $\alpha_{0,2}$ not both 0. The ideal of C-relations of such a sequence $\boldsymbol{u}$ is

$$\begin{aligned} J_{(a_1,a_2),3,(\alpha_{2,0},\alpha_{0,2})} = \big\langle & (x_1 - a_1)^3, \\ & \hat{a}_1^2(x_2 - a_2) - \beta_{1,2}\hat{a}_2(x_1 - a_1)^2 - \beta_{1,1}\hat{a}_1\hat{a}_2(x_1 - a_1), \\ & \hat{a}_2^2(x_1 - a_1) - \beta_{2,2}\hat{a}_1(x_2 - a_2)^2 - \beta_{2,1}\hat{a}_1\hat{a}_2(x_2 - a_2), \\ & (x_2 - a_2)^3 \big\rangle, \end{aligned}$$

where the $\beta_{i,j}$'s are polynomials in the $\alpha_{i,j}$'s with $1 \leq i + j \leq 2$. Furthermore, the $\beta_{i,j}$'s are such that either the first two or the last two polynomials span the ideal.

Geometrically, there exists another ideal whose only root is (a_1, a_2), and with multiplicity 3. This is $K_{(a_1,a_2),3} = \left\langle (x_1 - a_1)^2, (x_1 - a_1)(x_2 - a_2), (x_2 - a_2)^2 \right\rangle$.

Proposition A.2: Let $(a_1, a_2) \in \mathbb{K}^2$ and $K_{(a_1,a_2),3} = \left\langle (x_1 - a_1)^2, (x_1 - a_1)(x_2 - a_2), (x_2 - a_2)^2 \right\rangle$. Then, there is no sequence $\boldsymbol{u}$ such that $\mathrm{I_C}(\boldsymbol{u}) = K_{(a_1,a_2),3}$.

Proof. Let $\boldsymbol{u}$ be such that $K_{(a_1,a_2),3} \subseteq \mathrm{I_C}(\boldsymbol{u})$. Since $K_{(a_1,a_2),3}$ contains $\left\langle (x_1 - a_1)^2, (x_2 - a_2)^2 \right\rangle$, $\boldsymbol{u} = \mathfrak{u}_{a_1,e_1} \otimes \mathfrak{u}_{a_2,e_2}$ with $e_1 \leq 2$ and $e_2 \leq 2$. Assuming $a_1, a_2 \neq 0$, then there exist $\alpha, \beta, \gamma, \delta \in \mathbb{K}$ such that

$$\forall\, (i_1, i_2) \in \mathbb{N}^2, \quad u_{i_1,i_2} = (\alpha + \beta i_1 + \gamma i_2 + \delta i_1 i_2) a_1^{i_1} a_2^{i_2}.$$

Since $x_1x_2 - a_2x_1 - a_1x_2 + a_1a_2 = (x_1 - a_1)(x_2 - a_2) \in K_{(a_1,a_2),3}$,

$$\begin{aligned} u_{1,1} &= a_2 u_{1,0} + a_1 u_{0,1} - a_1 a_2 u_{0,0} \\ (\alpha + \beta + \gamma + \delta) a_1 a_2 &= (\alpha + \beta) a_1 a_2 + (\alpha + \gamma) a_1 a_2 - \alpha a_1 a_2 \\ \delta a_1 a_2 &= 0 \end{aligned}$$

and $\delta = 0$. Now, we saw that if $\gamma \neq 0$, then $\mathrm{I_C}(\boldsymbol{u}) \supseteq J_{(a_1,a_2),2,\beta/\gamma} = \left\langle x_1 - \frac{\beta}{\gamma} x_2 + \left(\frac{\beta}{\gamma} - 1\right), (x_2 - a_2)^2 \right\rangle \supsetneq K_{(a_1,a_2),3}$ and otherwise $\mathrm{I_C}(\boldsymbol{u}) \supseteq J_{(a_1,a_2),2,\infty} = \left\langle (x_1 - a_1)^2, x_2 - a_2 \right\rangle \supsetneq K_{(a_1,a_2),3}$.

If either $a_1 = 0$ or $a_2 = 0$, then this proof can be adapted using the appropriate sequence $\mathfrak{u}_{a_1,e_1} \otimes \mathfrak{u}_{a_2,e_2}$. This concludes the proof that no sequence has $K_{(a_1,a_2),3}$ as its ideal of C-relations. □

Bibliography

[1] J. Alman and V. Vassilevska Williams. A Refined Laser Method and Faster Matrix Multiplication. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 522–539, 2021.

[2] M. Bader and C. Zenger. Cache oblivious matrix multiplication using an element ordering based on a peano curve. *Linear Algebra and its Applications*, 417(2):301–313, 2006. Special Issue in honor of Friedrich Ludwig Bauer.

[3] C. Banderier and P. Flajolet. Basic analytic combinatorics of directed lattice paths. *Theoret. Comput. Sci.*, 281(1–2):37–80, 2002. Selected Papers in honour of Maurice Nivat.

[4] D. A. Bayer. *The Division Algorithm and the Hilbert Scheme*. PhD thesis, Harvard University, USA, 1982. AAI8222588.

[5] E. Becker, T. Mora, M. G. Marinari, and C. Traverso. The shape of the shape lemma. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, ISSAC '94, pages 129–133, New York, NY, USA, 1994. ACM.

[6] B. Beckermann and G. Labahn. A uniform approach for the fast computation of matrix-type pade approximants. *SIAM J. Matrix Anal. Appl.*, 15(3):804–823, 1994.

[7] M. R. Bender, J.-Ch. Faugère, and E. Tsigaridas. Towards mixed Gröbner basis algorithms: The multihomogeneous and sparse case. In *Proceedings of the 2018 ACM International Symposium on Symbolic and Algebraic Computation*, ISSAC '18, pages 71–78, New York, NY, USA, 2018. ACM.

[8] E. Berlekamp. Nonbinary BCH decoding. *IEEE Trans. Inform. Theory*, 14(2):242–242, 1968.

[9] L. Bernardin, P. Chin, P. Demarco, K. O. Geddes, D. E. G. Hare, K. M. Heal, G. Labahn, J. Mccarron, M. B. Monagan, D. Ohashi, and S. M. Vorkoetter. Maple programming guide, 1996.

[10] J. Berthomieu, A. Bostan, A. Ferguson, and M. Safey El Din. Gröbner bases and critical values: The asymptotic combinatorics of determinantal systems. *J. Algebra*, 602:154–180, 2022.

[11] J. Berthomieu, B. Boyer, and J.-Ch. Faugère. Linear algebra for computing gröbner bases of linear recursive multidimensional sequences. In *Proceedings of the 2015 ACM on International Symposium on Symbolic and Algebraic Computation*, ISSAC '15, pages 61–68, New York, NY, USA, 2015. ACM.

[12] J. Berthomieu, B. Boyer, and J.-Ch. Faugère. Linear Algebra for Computing Gröbner Bases of Linear Recursive Multidimensional Sequences. *Journal of Symbolic Computation*, 83(Supplement C):36–67, 2017. Special issue on the conference ISSAC 2015: Symbolic computation and computer algebra.

[13] J. Berthomieu, Ch. Eder, and M. Safey El Din. Msolve: A library for solving polynomial systems. In *Proceedings of the 2021 on International Symposium on Symbolic and Algebraic Computation*, ISSAC '21, pages 51–58, New York, NY, USA, 2021. Association for Computing Machinery.

[14] J. Berthomieu, Ch. Eder, and M. Safey El Din. msolve: A library for solving polynomial systems, 2021. https://msolve.lip6.fr/.

[15] J. Berthomieu, Ch. Eder, and M. Safey El Din. New efficient algorithms for computing Gröbner bases of saturation ideals (F4SAT) and colon ideals (Sparse-FGLM-colon). preprint, 2022.

[16] J. Berthomieu and J.-Ch. Faugère. Guessing Linear Recurrence Relations of Sequence Tuples and P-Recursive Sequences with Linear Algebra. In *Proceedings of the ACM on International Symposium on Symbolic and Algebraic Computation*, ISSAC '16, pages 95–102, New York, NY, USA, 2016. Association for Computing Machinery.

[17] J. Berthomieu and J.-Ch. Faugère. A polynomial-division-based algorithm for computing linear recurrence relations. In *Proceedings of the 2018 ACM International Symposium on Symbolic and Algebraic Computation*, ISSAC '18, pages 79–86, New York, NY, USA, 2018. ACM.

[18] J. Berthomieu and J.-Ch. Faugère. In-depth comparison of the Berlekamp–Massey–Sakata and the Scalar-FGLM algorithms: The adaptive variants. *Journal of Symbolic Computation*, 101:270–303, 2020.

[19] J. Berthomieu and J.-Ch. Faugère. Polynomial-division-based algorithms for computing linear recurrence relations. *Journal of Symbolic Computation*, 109:1–30, 2022.

[20] J. Berthomieu, V. Neiger, and M. Safey El Din. Faster Change of Order Algorithm for Gröbner Bases under Shape and Stability Assumptions. In *Proceedings of the 2022 International Symposium on Symbolic and Algebraic Computation*, ISSAC '22, pages 409–418, New York, NY, USA, 2022. Association for Computing Machinery.

[21] J. Berthomieu and M. Safey El Din. Guessing Gröbner bases of structured ideals of relations of sequences. *Journal of Symbolic Computation*, 111:1–26, 2022.

[22] P. Boito and O. Ruatta. Extended Companion Matrix for Approximate GCD. In *Proceedings of the 2011 International Workshop on Symbolic-Numeric Computation*, SNC '11, pages 74–80, New York, NY, USA, 2012. Association for Computing Machinery.

[23] R. Bose and D. Ray-Chaudhuri. On a class of error correcting binary group codes. *Information and Control*, 3(1):68–79, 1960.

[24] W. Bosma, J. Cannon, and C. Playoust. The Magma algebra system. I. The user language. *J. Symbolic Comput.*, 24(3-4):235–265, 1997. Computational algebra and number theory (London, 1993).

[25] A. Bostan, M. Bousquet-Mélou, M. Kauers, and S. Melczer. On 3-dimensional lattice walks confined to the positive octant. *Annals of Combinatorics*, 20(4):661–704, 2016.

[26] A. Bostan, C.-P. Jeannerod, C. Mouilleron, and E. Schost. On matrices with displacement structure: Generalized operators and faster algorithms. *SIAM Journal on Matrix Analysis and Applications*, 38(3):733–775, 2017.

[27] A. Bostan, C.-P. Jeannerod, and É. Schost. Solving Toeplitz- and Vandermonde-like Linear Systems with Large Displacement Rank. In C. W. Brown, editor, *ISSAC '07*, pages 33–40. ACM Press, 2007.

[28] A. Bostan, K. Raschel, and B. Salvy. Non-D-finite excursions in the quarter plane. *J. Combin. Theory Ser. A*, 121:45–63, 2014.

[29] A. Bostan, B. Salvy, and É. Schost. Fast Algorithms for Zero-Dimensional Polynomial Systems Using Duality. *Appl. Algebra Eng. Commun. Comput.*, 14(4):239–272, 2003.

[30] M. Bousquet-Mélou and M. Mishna. Walks with small steps in the quarter plane. In *Algorithmic probability and combinatorics*, volume 520 of *Contemp. Math.*, pages 1–39. Amer. Math. Soc., Providence, RI, 2010.

[31] M. Bousquet-Mélou and M. Petkovšek. Walks confined in a quadrant are not always d-finite. *Theoret. Comput. Sci.*, 307(2):257–276, 2003. Random Generation of Combinatorial Objects and Bijective Combinatorics.

[32] J. Brachat, P. Comon, B. Mourrain, and E. P. P. Tsigaridas. Symmetric tensor decomposition. *Linear Algebra Appl.*, 433(11-12):1851–1872, 2010.

[33] P. Breiding, B. Sturmfels, and S. Timme. 3264 conics in a second. *Notices of the American Mathematical Society*, 67(1):30–37, 2020.

[34] R. P. Brent, F. G. Gustavson, and D. Y. Yun. Fast solution of Toeplitz systems of equations and computation of Padé approximants. *Journal of Algorithms*, 1(3):259–295, 1980.

[35] B. Buchberger. *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal.* PhD thesis, University of Innsbruck, 1965.

[36] B. Buchberger. A Criterion for Detecting Unnecessary Reductions in the Construction of Gröbner Bases. In *EUROSAM '79, An International Symposium on Symbolic and Algebraic Manipulation*, volume 72 of *Lecture Notes in Computer Science*, pages 3–21, Berlin, 1979. Springer.

[37] L. Caniglia, A. Galligo, and J. Heintz. Some New Effectivity Bounds in Computational Geometry. In *Proceedings of the 6th International Conference, on Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, AAECC-6, pages 131–151, Berlin, Heidelberg, 1988. Springer-Verlag.

[38] S. Collart, M. Kalkbrener, and D. Mall. Converting Bases with the Gröbner Walk. *J. Symbolic Comput.*, 24(3):465–469, 1997.

[39] C. Conradi, E. Feliu, M. Mincheva, and C. Wiuf. Identifying parameter regions for multistationarity. *PLOS Computational Biology*, 13(10):e1005751, 2017.

[40] D. Coppersmith. Solving homogeneous linear equations over GF(2) via block Wiedemann algorithm. *Math. Comp.*, 62(205):333–350, 1994.

[41] D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties, and Algorithms.* Undergraduate Texts in Mathematics. Springer, New York, fourth edition, 2015. An introduction to computational algebraic geometry and commutative algebra.

[42] W. Decker, G.-M. Greuel, G. Pfister, and H. Schönemann. Singular 4-1-2 — A computer algebra system for polynomial computations. http://www.singular.uni-kl.de, 2019.

[43] R. Duan, H. Wu, and R. Zhou. Faster Matrix Multiplication via Asymmetric Hashing, 2022.

[44] Ch. Eder and J.-Ch. Faugère. A survey on signature-based algorithms for computing gröbner bases. *Journal of Symbolic Computation*, 80:719–784, 2017.

[45] M. Elkadi and B. Mourrain. *Introduction à la résolution des systèmes polynomiaux*, volume 59 of *Mathématiques et Applications.* Springer, 2007.

[46] J.-C. Faugère, P. Gaudry, L. Huot, and G. Renault. Sub-Cubic Change of Ordering for Gröbner Basis: A Probabilistic Approach. In *Proceedings ISSAC 2014*, pages 170–177. ACM, 2014.

[47] J.-Ch. Faugère. A New Efficient Algorithm for Computing Gröbner bases (F4). *Journal of Pure and Applied Algebra*, 139(1):61–88, 1999.

[48] J.-Ch. Faugère. A New Efficient Algorithm for Computing Gröbner Bases Without Reduction to Zero (F5). In *Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation*, ISSAC '02, pages 75–83, New York, NY, USA, 2002. ACM.

[49] J.-Ch. Faugère. FGb: A Library for Computing Gröbner Bases. In K. Fukuda, J. v. d. Hoeven, M. Joswig, and N. Takayama, editors, *Mathematical Software – ICMS 2010*, pages 84–87, Berlin, Heidelberg, 2010. Springer.

[50] J.-Ch. Faugère, P. Gianni, D. Lazard, and T. Mora. Efficient Computation of Zero-dimensional Gröbner Bases by Change of Ordering. *J. Symbolic Comput.*, 16(4):329–344, 1993.

[51] J.-Ch. Faugère and A. Joux. Algebraic Cryptanalysis of Hidden Field Equation (HFE) Cryptosystems Using Gröbner Bases. In D. Boneh, editor, *Advances in Cryptology - CRYPTO 2003*, Lecture Notes in Computer Science, pages 44–60, Berlin, Heidelberg, 2003. Springer.

[52] J.-Ch. Faugère and C. Mou. Fast algorithm for change of ordering of zero-dimensional gröbner bases with sparse multiplication matrices. In *Proceedings of the 36th International Symposium on Symbolic and Algebraic Computation*, ISSAC '11, pages 115–122, New York, NY, USA, 2011. ACM.

[53] J.-Ch. Faugère and C. Mou. Sparse FGLM algorithms. *Journal of Symbolic Computation*, 80(3):538–569, 2017.

[54] J.-Ch. Faugère, M. Safey El Din, and Th. Verron. On the Complexity of Computing Gröbner Bases for Quasi-Homogeneous Systems. In *Proceedings of the 38th International Symposium on Symbolic and Algebraic Computation*, ISSAC '13, pages 189–196, New York, NY, USA, 2013. Association for Computing Machinery.

[55] J.-Ch. Faugère, M. Safey El Din, and Th. Verron. On the complexity of computing Gröbner bases for weighted homogeneous systems. *Journal of Symbolic Computation*, 76:107–141, 2016.

[56] J.-Ch. Faugère, P.-J. Spaenlehauer, and J. Svartz. Sparse Gröbner bases: The unmixed case. In *Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation*, ISSAC '14, pages 178–185, New York, NY, USA, 2014. ACM.

[57] J.-Ch. Faugère and J. Svartz. Gröbner bases of ideals invariant under a commutative group: The non-modular case. In *Proceedings of the 38th International Symposium on Symbolic and Algebraic Computation*, ISSAC '13, pages 347–354, New York, NY, USA, 2013. ACM.

[58] A. Ferguson and H. P. Le. Finer complexity estimates for the change of ordering of gröbner bases for generic symmetric determinantal ideals. In *Proceedings of the 2022 International Symposium on Symbolic and Algebraic Computation*, ISSAC '22, pages 399–407, New York, NY, USA, 2022. Association for Computing Machinery.

[59] P. Fortin, M. Gouicem, and S. Graillat. GPU-Accelerated Generation of Correctly Rounded Elementary Functions. *ACM Trans. Math. Softw.*, 43(3), 2016.

[60] J. García Fontán, A. Nayak, and M. Briot, S. Safey El Din. Singularity Analysis for the Perspective-Four and Five-Line Problems. *International Journal of Computer Vision*, 2022.

[61] M. R. Garey and D. S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., USA, 1990.

[62] K. Gatermann and B. Huber. A Family of Sparse Polynomial Systems Arising in Chemical Reaction Systems. *Journal of Symbolic Computation*, 33(3):275–305, 2002.

[63] D. Gorenstein. An arithmetic theory of adjoint plane curves. *Trans. Amer. Math. Soc.*, 72:414–436, 1952.

[64] D. Henrion, J.-B. Lasserre, and C. Savorgnan. Approximate Volume and Integration for Basic Semialgebraic Sets. *SIAM Review*, 51(4):722–743, 2009.

[65] A. Hocquenghem. Codes correcteurs d'erreurs. *Chiffres*, 2:147–156, 1959.

[66] S. G. Hyun, V. Neiger, H. Rahkooy, and É. Schost. Block-Krylov techniques in the context of sparse-FGLM algorithms. *Journal of Symbolic Computation*, 98:163–191, 2020. Special Issue on Symb. and Alg. Comp.: ISSAC 2017.

[67] W. R. Inc. Mathematica, Version 13.1. Champaign, IL, 2022.

[68] C.-P. Jeannerod, V. Neiger, and G. Villard. Fast computation of approximant bases in canonical form. *Journal of Symbolic Computation*, 98:192–224, 2020. Special Issue on Symbolic and Algebraic Computation: ISSAC 2017.

[69] M. Kauers, M. Jaroschek, and F. Johansson. Ore Polynomials in Sage. In J. Gutierrez, J. Schicho, and M. Weimann, editors, *Computer Algebra and Polynomials: Applications of Algebra and Number Theory*, pages 105–125, Cham, 2015. Springer International Publishing.

[70] C. Kollreider and B. Buchberger. An improved algorithmic construction of Gröbner-bases for polynomial ideals. *SIGSAM Bull.*, 12:27–36, 1978.

[71] Ch. Koutschan. Holonomicfunctions (user's guide). Technical Report 10-01, Johannes Kepler Universitä Linz, 2010.

[72] Y. Kuang, Y. Zheng, and K. øAström. Partial Symmetry in Polynomial Systems and Its Applications in Computer Vision. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 438–445, 2014.

[73] P. Lairez and M. Safey El Din. Computing the dimension of real algebraic sets. In *Proceedings of the 46th International Symposium on Symbolic and Algebraic Computation*, ISSAC '21, pages 257–264, New York, NY, USA, 2021. Association for Computing Machinery.

[74] D. Lazard. Gröbner bases, Gaussian elimination and resolution of systems of algebraic equations. In *Computer algebra (London, 1983)*, volume 162 of *Lecture Notes in Comput. Sci.*, pages 146–156. Springer, Berlin, 1983.

[75] V. Levandovskyy. *Non-commutative Computer Algebra for polynomial algebras: Gröbner bases, applications and implementation.* PhD thesis, Technische Universität Kaiserslautern, 2005.

[76] F. S. Macaulay. Modern algebra and polynomial ideals. *Mathematical Proceedings of the Cambridge Philosophical Society*, 30:27–46, 1934.

[77] J. L. Massey. Shift-register synthesis and BCH decoding. *IEEE Trans. Inform. Theory*, IT-15:122–127, 1969.

[78] S.-J. Matsubara-Heo and S. Telen. Twisted cohomology and likelihood ideals, 2023.

[79] H. M. Möller and F. Mora. Upper and lower bounds for the degree of groebner bases. In J. Fitch, editor, *EUROSAM 84*, pages 172–183, Berlin, Heidelberg, 1984. Springer Berlin Heidelberg.

[80] G. Moreno-Socías. Degrevlex Gröbner bases of generic complete intersections. *Journal of Pure and Applied Algebra*, 180(3):263–283, 2003.

[81] B. Mourrain. Fast algorithm for border bases of artinian gorenstein algebras. In *Proceedings of the 2017 ACM on International Symposium on Symbolic and Algebraic Computation*, ISSAC '17, pages 333–340, New York, NY, USA, 2017. ACM.

[82] S. Naldi and V. Neiger. A divide-and-conquer algorithm for computing gröbner bases of syzygies in finite dimension. In *Proceedings of the 45th International Symposium on Symbolic and Algebraic Computation*, ISSAC '20, pages 380–387, New York, NY, USA, 2020. Association for Computing Machinery.

[83] V. Neiger and É. Schost. Computing syzygies in finite dimension using fast linear algebra. *Journal of Complexity*, 60:101502, 2020.

[84] J. Nie and K. Ranestad. Algebraic degree of polynomial optimization. *SIAM J. Optim.*, 20(1):485–502, 2009.

[85] S. Odake and R. Sasaki. Exactly solvable quantum mechanics and infinite families of multi-indexed orthogonal polynomials. *Physics Letters B*, 702(2):164–170, 2011.

[86] B. Parisse and R. De Graeve. Giac/Xcas, version 1.5.0, 2018. http://www-fourier.univ-grenoble-alpes.fr/~parisse/giac.html.

[87] B. Pascual-Escudero, A. Nayak, S. Briot, O. Kermorgant, Ph. Martinet, M. Safey El Din, and F. Chaumette. Complete Singularity Analysis for the Perspective-Four-Point Problem. *International Journal of Computer Vision*, 129(4):1217–1237, 2021.

[88] C. Pernet and A. Storjohann. Faster Algorithms for the Characteristic Polynomial. In *Proceedings ISSAC 2007*, pages 307–314. ACM, 2007.

[89] C. Pernet and A. Storjohann. Frobenius form in expected matrix multiplication time over sufficiently large fields. unpublished report, 2007.

[90] J. Rabinowitsch. Zum Hilbertschen Nullstellensatz. *Mathematische Annalen*, 102:520–520, 1930.

[91] M. Raghavan and B. Roth. Solving Polynomial Systems for the Kinematic Analysis and Synthesis of Mechanisms and Robot Manipulators. *Journal of Mechanical Design*, 117(B):71–79, 1995.

[92] M. Safey El Din. Computing Sampling Points on a Singular Real Hypersurface using Lagrange's System. Research Report RR-5464, INRIA, 2005.

[93] S. Sakata. Finding a minimal set of linear recurring relations capable of generating a given finite two-dimensional array. *J. Symbolic Comput.*, 5(3):321–337, 1988.

[94] S. Sakata. N-dimensional Berlekamp-Massey algorithm for multiple arrays and construction of multivariate polynomials with preassigned zeros. In T. Mora, editor, *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, volume 357 of *Lecture Notes in Computer Science*, pages 356–376. Springer Berlin Heidelberg, 1989.

[95] S. Sakata. Extension of the Berlekamp-Massey algorithm to N Dimensions. *Inform. and Comput.*, 84(2):207–239, 1990.

[96] S. Sakata. Decoding binary 2-D cyclic codes by the 2-D Berlekamp-Massey algorithm. *IEEE Trans. Inform. Theory*, 37(4):1200–1203, 1991.

[97] S. Sakata. The bms algorithm. In M. Sala, S. Sakata, T. Mora, C. Traverso, and L. Perret, editors, *Gröbner Bases, Coding, and Cryptography*, pages 143–163. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[98] B. Salvy. Linear Differential Equations as a Data Structure. *Foundations of Computational Mathematics*, 19:1071–1112, 2019.

[99] B. Salvy and P. Zimmermann. GFUN: A Maple Package for the Manipulation of Generating and Holonomic Functions in One Variable. *ACM Trans. Math. Softw.*, 20(2):163–177, jun 1994.

[100] T. Sauer. Prony's method in several variables: Symbolic solutions by universal interpolation. *Journal of Symbolic Computation*, 84:95–112, 2018.

[101] A. K. Steel. Direct solution of the (11, 9, 8)-MinRank problem by the block Wiedemann algorithm in magma with a tesla GPU. In *Proceedings of the 2015 International Workshop on Parallel Symbolic Computation, PASCO 2015, Bath, United Kingdom, July 10-12, 2015*, pages 2–6. ACM, 2015.

[102] A. Storjohann. *Algorithms for Matrix Canonical Forms.* Phd thesis, Swiss Federal Institute of Technology – ETH, 2000.

[103] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 9.1)*, 2020. https://www.sagemath.org.

[104] C. Traverso. Hilbert Functions and the Buchberger Algorithm. *Journal of Symbolic Computation*, 22(4):355–376, 1996.

[105] Th. Verron. On the computation of Gröbner bases for matrix-weighted homogeneous systems, 2022.

[106] D. Wiedemann. Solving sparse linear equations over finite fields. *IEEE Trans. Inf. Theory*, 32(1):54–62, 1986.

Abstract

This habilitation thesis deals with polynomial system solving through Gröbner bases computations. It focuses on the link between multivariate polynomials and linear recurrence relations satisfied by a multi-indexed sequence for computing Gröbner bases.

Our contributions mainly lie on the theoretical and practical aspects on these Gröbner bases computations. First, we present `msolve`, a new open source `C` library, for solving polynomial systems using Gröbner bases. Second, we describe new algorithms and complexity estimates for computing Gröbner bases either for a total degree order or the lexicographic one. Then, we present linear algebras-based and polynomial-division-based algorithms for guessing linear recurrences with constant or polynomial coefficients, in generic and structured situations.

Finally, we detail our research project for the forthcoming years on these aspects.

Résumé

Cette thèse d'habilitation traite de la résolution de systèmes polynomiaux *via* le calcul de bases de Gröbner. Elle se concentre sur le lien entre les polynômes multivariés et les relations de récurrence linéaires satisfaites par une suite multi-indexée pour calculer des bases de Gröbner.

Nos contributions portent principalement sur les aspects théoriques et pratiques de ces calculs de bases de Gröbner. Tout d'abord, nous présentons `msolve`, une nouvelle bibliothèque `C` *open source*, pour la résolution de systèmes polynomiaux en utilisant les bases de Gröbner. Ensuite, nous décrivons de nouveaux algorithmes et donnons des estimations de complexité pour le calcul des bases de Gröbner soit pour un ordre de degré total, soit pour l'ordre lexicographique. Ensuite, nous présentons des algorithmes basés sur l'algèbre linéaire et sur la division de polynômes pour deviner les récurrences linéaires à coefficients constants ou polynomiaux, dans des situations génériques et structurées.

Enfin, nous détaillons notre projet de recherche pour les années à venir sur ces aspects.